514-3SUG/0195

MISSION OPERATIONS AND DATA SYSTEMS DIRECTORATE

Landsat 7 Processing System (LPS) Users Guide for Release 1

February 1997

National Aeronautics and Space Administration

Goddard Space Flight Center
Greenbelt, Maryland

Landsat 7 Processing System (LPS)
Users Guide for Release 1

February 1997

Prepared by:

Dennis Crehan, Systems Engineering Date
Landsat 7 Processing System, CNMOS, Lockheed Martin

Concurred by:

Jeffrey Hosler, Software Engineering Manager Date
Landsat 7 Processing System, Code 514, Goddard Space Flight Center

Approved by:

Daniel Ferry, Software Task Leader Date
Landsat 7 Processing System, CNMOS, Computer Sciences Corporation

Approved by:

Nathaniel Daniel, Project Manager Date
Landsat 7 Processing System, CNMOS, Computer Sciences Corporation

Approved by:

Robert Schweiss, System Engineering Manager Date
Landsat 7 Processing System, Code 514, Goddard Space Flight Center

Approved by:

Joy Henegar, Project Manager Date
Landsat 7 Processing System, Code 514, Goddard Space Flight Center

Goddard Space Flight Center
Greenbelt, Maryland

Preface

This document is under the control of the LPS Central Review Board (CRB).

Configuration change requests (CCRs) to this document, as well as supportive material justifying the proposed change, shall be submitted to the LPS CRB. Changes to this document shall be made by document change notice (DCN) or by complete revision.

Address questions and proposed changes concerning this document to

Jeffrey C. Hosler, Software Manager
Landsat 7 Processing System
Code 514
Goddard Space Flight Center
Greenbelt, Maryland 20771

Abstract

This document provides instructions for operating the Landsat 7 Processing System (LPS). It instructs the novice user in the performance of specific LPS operations and provides a detailed reference for the experienced user.

Keywords: *Landsat 7, Landsat 7 Processing System (LPS), users guide*

Contents

1. Introduction

2. Getting Started

3. Data Capture Operations

4. Raw Wideband Data Management

5. Level 0R Processing Management

6. Files Transfers to EDC DAAC

7. Report Generation

8. Testing the LPS

9. LPS Maintenance

10. LPS GUI Detailed Reference

Appendix A. LPS Error Messages

Tables

Figures

1. Introduction

1.1 Purpose and Scope

This document provides instructions for operating the Landsat 7 Processing System (LPS). It instructs the novice user in the performance of specific operations and provides a detailed reference for the experienced user.

This users guide covers the following topics:

- Logging onto the LPS system hosts
- Setting up one's user environment
- Starting up and shutting down the LPS software
- Navigating the LPS graphical user interface (GUI)
- Monitoring LPS status
- Performing basic operations with the LPS
- Detailed LPS GUI menu options reference
- LPS utilities
- LPS troubleshooting

This guide assumes the reader is familiar with point-and-click interfaces that use a mouse, buttons, pulldown menus, and dialogs (or forms) and with the fundamentals of interacting with one of the UNIX shells (csh or sh). This guide also assumes familiarity with LPS standard operating procedures.

This guide does not provide instructions for third-party software such as Oracle's Database Management System (DBMS) or the Generic Telemetry Simulator (GTSIM). The reader should consult vendor-supplied documentation for those tools. This guide also does not provide instructions for LPS software maintenance. The reader should consult the *LPS Programmers Reference Manual* (Reference 1).

1.2 Organization

This document contains 10 chapters and 6 appendixes:

- Chapter 1, *Introduction*, provides an introduction to this document and an overview of the LPS.
- Chapter 2, *Getting Started*, provides instructions for logging on to the LPS, setting up an environment, starting up and shutting down the LPS software, navigating the LPS operations interface, and monitoring LPS status.
- Chapter 3, *Data Capture Operations*, provides instructions for managing and monitoring automatic data capture and for performing data capture manually.

- Chapter 4, *Raw Wideband Data Management*, provides instructions for managing and monitoring the automatic movement of captured raw wideband data to the 30-day store and moving captured raw wideband data files between LPS disks and the removable media of the 30-day store manually.
- Chapter 5, *Level 0R Processing*, provides instructions for managing automatic Level Zero R (L0R) processing, performing L0R processing manually, monitoring L0R processing, and modifying L0R processing parameters and error-reporting thresholds.
- Chapter 6, *File Transfers to EDC DAAC*, provides instructions for managing and monitoring the automatic transfer of LPS output files to the Earth Resources Observation System (EROS) Data Center (EDC) Distributed Active Archive Center (DAAC) and for performing the transfer manually.
- Chapter 7, *Report Generation*, provides instructions for generating LPS reports.
- Chapter 8, *Testing the LPS*, provides instructions for performing LPS built-in tests.
- Chapter 9, *LPS Maintenance*, provides instructions for using assorted LPS utility programs.
- Chapter 10, *LPS GUI Detailed Reference*, provides a detailed explanation of all menu options in the LPS GUI.
- Appendix A, *LPS Error Messages*, describes all error messages output by the LPS.
- Appendix B, *LPS Directory Structure and File Name Formats*, explains the directory hierarchy on LPS strings and the format of LPS file names.
- Appendix C, *Process Catalog*, describes all LPS processes.
- Appendix D, *Customizing Your Environment*, explains what environment variables are important for LPS operations and how to customize their values.
- Appendix E, *Invoking LPS Programs From IRIX*, explains how to execute LPS programs and use IRIX and ORACLE features to perform LPS operations from an IRIX shell. Expert users can use this information to construct scripts that extend the capabilities of the LPS.
- Appendix F, *Man Pages for LPS Standalone Programs*, contains UNIX-style man pages for all LPS programs that can be invoked from an IRIX shell.

1.3 LPS Overview

This section describes the LPS environment, functions of the LPS, its hardware configuration, the components of its software, and the operator's role in LPS operations.

1.3.1 Environment

The LPS is a component of the Landsat 7 System. The Landsat 7 System provides wide-area, multi-spectral imaging of the Earth. Landsat 7 data can be processed into images or used in a variety of scientific, military, and commercial applications. The Landsat 7 System consists of flight components (including the Landsat 7 satellite) and ground components (including the

LPS). Landsat 7 flight components are responsible for launching, placing, and maintaining the Landsat 7 satellite. Landsat 7 ground components are responsible for operating the satellite and for receiving, processing, archiving, and distributing Earth observation data. Figure 1–1 shows the elements of the Landsat 7 System that are important to LPS operations.

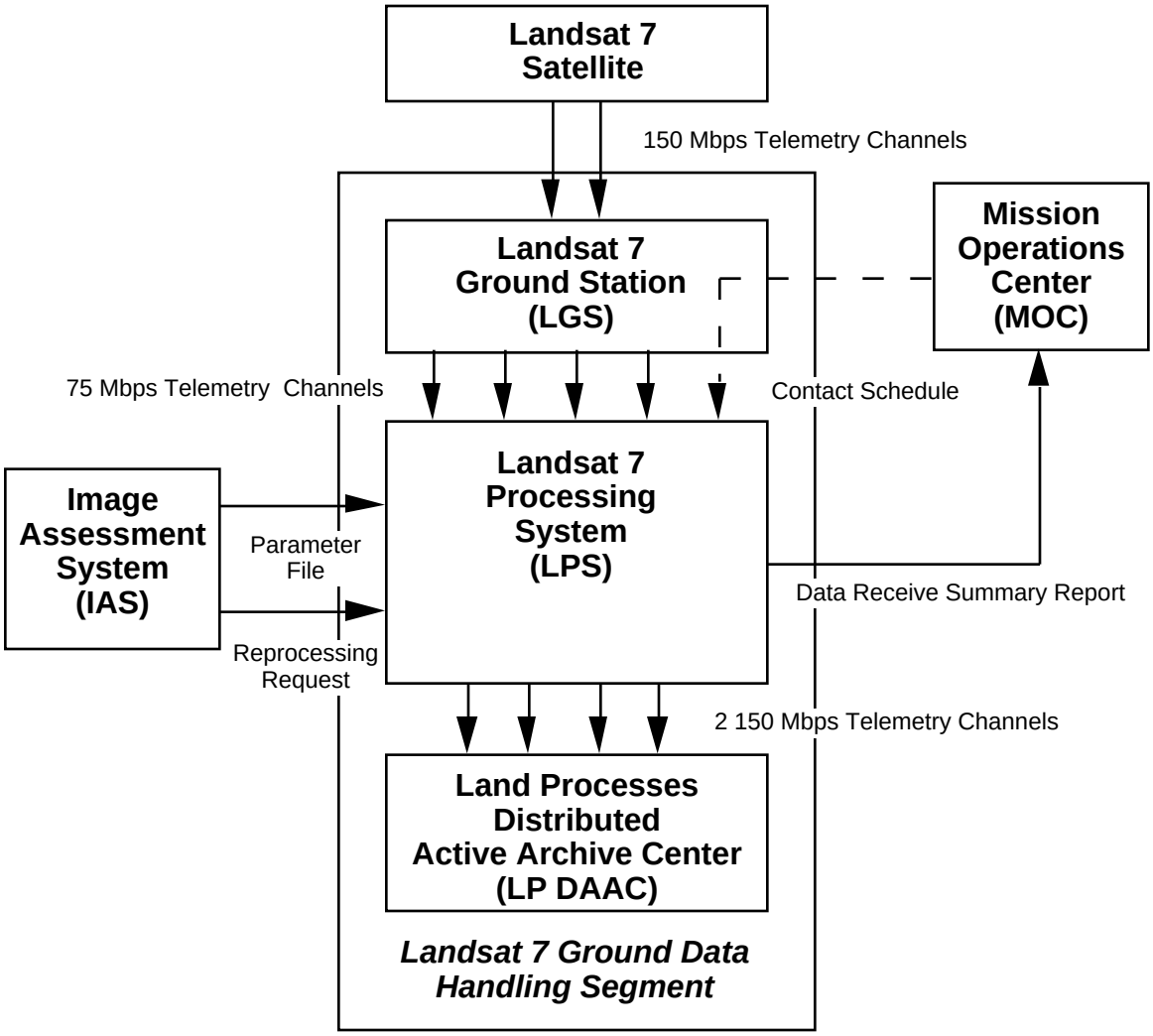

Figure 1–1. The LPS Environment

The Landsat 7 satellite is in a Sun-synchronous orbit along a track defined by the Worldwide Reference System (WRS). The WRS divides the Earth into 57,784 scenes, each representing a swath traversed by the satellite in 23.92 seconds. The Landsat 7 satellite's flight repeats the coverage of all WRS scenes every 16 days.

Onboard the Landsat 7 satellite is the Enhanced Thematic Mapper Plus (ETM+), a fixed-position, nadir-viewing instrument. Its viewing swath is produced by a mirror that sweeps across track as the sensor field of view moves forward due to satellite motion. Calibration data and mirror scan correction data (MSCD) are collected at the end of each scan-mirror cycle. The ETM+ collects data in eight spectral bands. Bands 1, 2, 3, 4, and 8 collect the visible near

infrared spectral range. Bands 5 and 7 are short wavelength infrared bands. Band 6 is a thermal long wavelength infrared band.

The ETM+ provides output data via two 75 megabits per second (Mbps) channels. Channel I contains multiplexed data from Bands 1, 2, 3, 4, 5, and 6. Channel Q contains multiplexed data from Bands 6, 7, and 8. Band 6 data is duplicated in both channels. Both channels also contain ETM+ status and payload correction data (PCD) associated with the image data. The Landsat 7 satellite uses one of two onboard solid state recorders to capture imaging data when it cannot be transmitted to the ground in real time.

When a ground station is in sight, Landsat 7 uses two of its three 150-Mbps X-band direct downlinks to simultaneously transmit both real-time and recorded image wideband data to the ground station. Several different segments of image wideband data collected at different times and locations, called subintervals, are normally transmitted in succession. Each segment contains continuous imagery for one or more WRS scenes. Because the satellite can play back any portion of the data recorded in its onboard recorders in any order, the acquisition times and locations of the scenes in the subintervals transmitted during a contact have no particular relation to one another.

The Landsat Ground Station (LGS) acquires ETM+ wideband data directly from Landsat 7 via two 150-Mbps X-band return links, separates each X band into two 75-Mbps I and Q channels, and transmits the acquired wideband data over four 75-Mbps LGS output channels to the LPS. The LGS receives Landsat 7 contact period schedules from the Landsat 7 Mission Operations Center (MOC) and forwards them electronically to the LPS, which uses them to schedule the start and stop of its capture of raw wideband data. After each contact, LPS operations personnel generate a Data Receive Summary Report, describing the data captured, and forward it to the MOC, either through voice link or facsimile transmission.

After a contact, the LPS processes the captured raw wideband data to L0R (described in Section 1.3.2) and automatically notifies EDC DAAC that files are available. EDC DAAC transfers the output files and archives them. Landsat 7 data users access EDC DAAC to query the archive for available data and to order data products.

The Image Assessment System (IAS) maintains Landsat 7 data quality. The IAS reviews the quality of a subset of Landsat 7 data. When problems are discovered, IAS operations personnel may request that data be reprocessed. IAS operations personnel specify the contact period for reprocessing to LPS operations personnel. LPS operators restage the raw wideband data from tape to disk and initiate its reprocessing. The resulting output files are then automatically transferred to EDC DAAC.

1.3.2 Functions

The LPS's principal functions are capturing raw wideband data from the Landsat 7 satellite, processing the captured data to L0R, and cooperating in the transfer of the L0R data files to EDC DAAC. The following subsections describe each of these functions in more detail.

1.3.2.1 Raw Wideband Data Capture

The LPS captures raw wideband data transmitted from the Landsat 7 satellite onto a 32-gigabyte (GB) storage array. Each operational LPS string captures one of the four 75-Mbps channels

received from the LGS. LPS software captures raw wideband data automatically, based on a contact schedule received from the LGS loaded into the database on operator command. The schedule indicates contact periods for both 150-Mbps downlinks from the Landsat 7 satellite. A capture source field in the LPS database that can be set by an operator indicates the channel to which a string is connected. A daemon process (mac_autocapture) on each LPS string monitors the schedule. It begins data capture shortly before scheduled acquisition of signal for the channel to which the daemon's LPS string is connected. Capture stops automatically at the scheduled time for loss of signal.

Everything the LPS receives through the LGS channel during the scheduled time of contact is written to a standard UNIX file. Because each string creates a separate file, four files are created for each contact. What the LPS receives can include noise as well as actual wideband data. Noise is received whenever there is a dropout (a temporary loss of the satellite's transmission) or when any other error in the communications path occurs causing an incorrect value to be received by the LPS.

When a contact is over, each LPS string automatically copies the newly created raw wideband data file to a Digital Linear Tape (DLT™). Because a second contact can occur while one contact's file is being copied to tape, the LPS software will wait for the tape drive to become available and, if necessary, for a new tape to be inserted.

1.3.2.2 L0R Processing

When a contact is over, each LPS string automatically begins processing the raw wideband data to L0R. Because a second (and indeed a third) contact can occur while one contact's file is being processed to L0R, the LPS software will wait until L0R processing on the first contact has completed before beginning work on subsequent contacts.

L0R processing is a multi-step process that locates and extracts ETM+ imagery and ancillary data from the raw wideband data stream captured during a contact to produce a number of output files written to a 32-GB storage array, separate from the storage array used to hold raw data.

The L0R processing steps are as follows:

- *Raw Data Processing* – LPS software extracts transfer frames from the raw wideband data file, applies a number of error detection and correction schemes to remove transmission errors, and annotates the transfer frame with a number of quality indicators before passing it on for further processing.
- *Major Frame Processing* – LPS software reconstructs ETM+ major frames from data extracted from the transfer frames, identifies the subintervals making up the contact, extracts image data for each of the eight spectral bands, and creates output files containing the MSCD and calibration data for each subinterval.
- *Image Data Processing* – LPS software takes the extracted image data and, for each subinterval, creates output files for each spectral band (in the case of Band 8, as many as four output files can be created for a single subinterval), produces browse files containing a reduced resolution image for each scene in the subinterval, automatically assesses the cloud cover for each scene, and provides a Moving Window Display (MWD) of the imagery being processed.
- *PCD Processing* – LPS software reconstructs PCD major frames from data extracted from transfer frames. LPS software also extracts the attitude and ephemeris data contained in each PCD major frame to determine the time at which the satellite passed over WRS scene centers. The scene center times are used to divide the imagery into scenes during image data processing.
- *Metadata File Generation* – LPS software generates a metadata file for each subinterval. The metadata provides information on the identification and quality of the L0R instrument data contained in a subinterval. Metadata also includes return-link data quality and ETM+ instrument data quality statistics, quality and accounting (Q&A) information on the received and processed PCD, and Cloud Cover Assessment (CCA) on both WRS scene and quadrant bases.

1.3.2.3 Output File Transfer

When L0R processing has completed successfully, the LPS software on each string automatically notifies EDC DAAC of its availability by sending a data availability notice (DAN). On receipt of this notification, EDC DAAC acknowledges the DAN by returning a data availability acknowledgment (DAA). Later, EDC DAAC transfers the available files from the LPS to its data archive storage using File Transfer Protocol (FTP). EDC DAAC notifies the LPS that it has attempted to transfer the files by sending a data delivery notice (DDN) describing the disposition of the transfer attempts. The LPS acknowledges the DDN by returning a data delivery acknowledgment (DDA).

If the disposition in a DDN indicates that a file has been successfully transferred, the LPS automatically deletes the file. Otherwise, the LPS outputs messages indicating that the transfer attempt failed. Files can be marked for retention. In this case, the output file is not deleted automatically. Files can be manually deleted at any time.

1.3.3 Hardware Configuration

The LPS hardware configuration for Release 1 is illustrated in Figure 1–2.

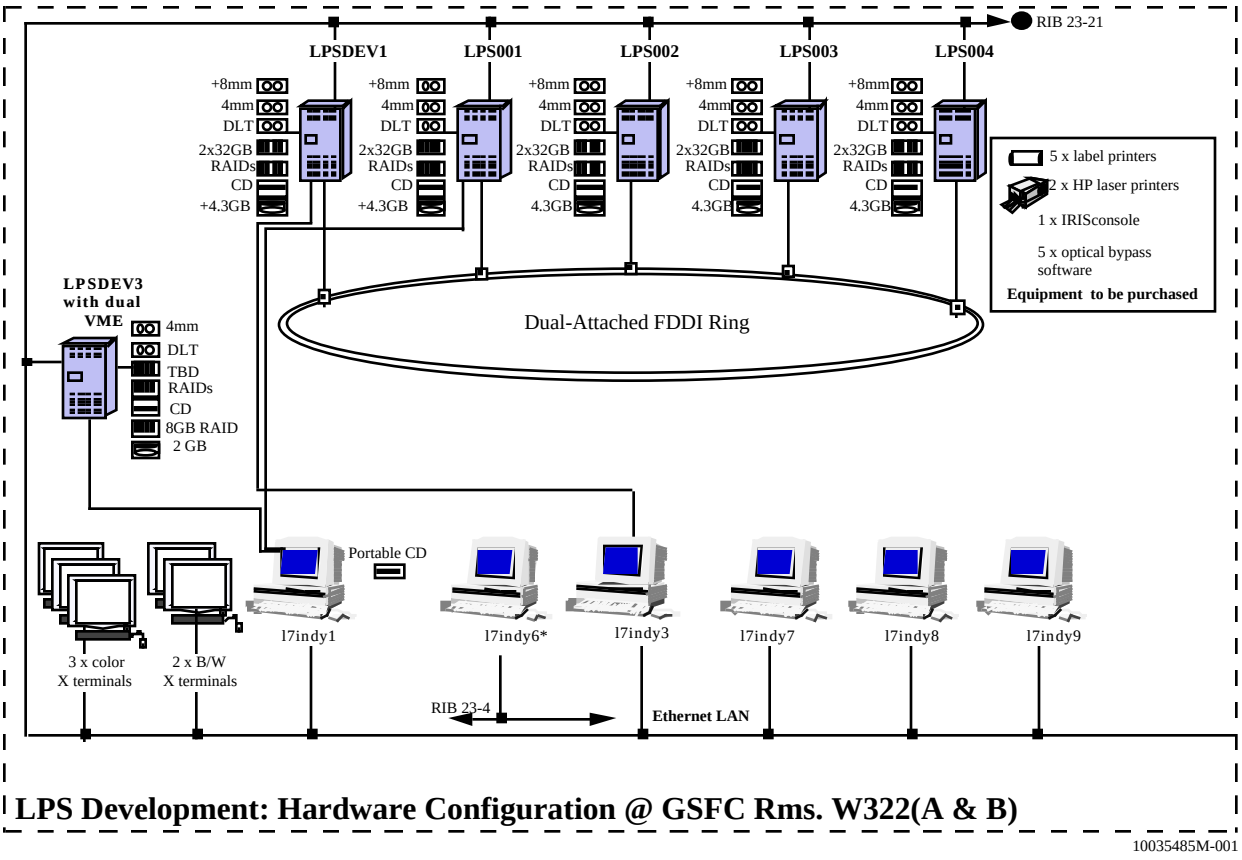

Figure 1–2. LPS Release 1 Hardware Configuration

The configuration includes the following components:

- *LPS Strings* – These are the hosts for nearly all LPS processing. The host for each string is a Silicon Graphics, Inc. (SGI) Challenge XL computer. Each string has eight processors and 512 megabytes (MB) of random access memory (RAM). Each string contains LGS output channel interface hardware through which the string can capture raw wideband data and a 32-GB storage array on which it can store captured data. For Release 1, the LGS interface hardware of one LPS string can be connected to any other to simulate the LGS-LPS interface. Another 32-GB storage array holds the output files produced by L0R processing. A 4-GB disk stores all other files, including the LPS executables and the LPS database. Each string hosts a DLT™ drive used to save raw wideband data to tape and to restage it to disk when reprocessing is requested. Each string also hosts a CD-ROM drive, a 4-millimeter tape drive, and an 8-millimeter tape drive. LPS strings are connected to each other and to Indy workstations and X terminals by an Ethernet network. A fiber-distributed digital interface (FDDI) network serves as the interface to EDC DAAC. For Release 1, the FDDI network interconnects the strings

so that one can serve as an EDC DAAC simulator. Finally, each LPS string has **TBD** serial connections to Indy workstations that serve as the string's console terminal.

- *Indy Workstations* – SGI Indy workstations are the hosts for the LPS MWD and also host the LPS string console terminals. They may be used to display the LPS software's GUI for one or more strings.

- *X Terminals* – These are the display devices for the LPS software's GUI. Each string has its own GUI, and a single X terminal can display the GUI from any number of strings. Normally, a single X terminal displays the GUI for two strings.

1.3.4 Software Components

LPS functions are performed by a number of software components, most of which are LPS application software written specifically for the LPS. However, a number of components are general-purpose software. The programs fall into three categories:

1. A collection of separate LPS standalone programs that perform specific LPS operations, such as capturing raw wideband data or processing a specified file to L0R

2. Daemon processes that execute in the background to control the automated LPS functions of data capture, copying to tape, and L0R processing

3. A GUI that allows the user to control and monitor the execution of the LPS programs

A basic feature of the LPS software is its division into functional and managing components. The standalone programs (category 1) perform specific LPS operations. The managing components (categories 2 and 3) do not perform operations. Rather, they cause the operations to occur at the appropriate time.

Figure 1–3 illustrates the LPS software's basic structure. Detailed descriptions of LPS processes appear in Appendix C. Once the LPS software has started, the LPS GUI and daemon processes are always active. Each command issued through the LPS GUI causes it to execute one (or more) of the LPS functional programs to perform the operation requested. The daemon processes are always active in the background. Each of them is waiting for an event. When the event occurs, the daemon starts up a functional program to perform the necessary processing. L0R processing is executed by a number of processes. Each process performs one of the steps necessary to transform raw wideband data into LPS output files.

LPS functional programs normally are invoked through the LPS GUI. However, all LPS functional programs may also be invoked directly from any IRIX shell (e.g., csh or sh). Invoke them, as for any IRIX program, by typing the program name and any parameters as a command to a shell. Commands can be included in shell scripts.

By writing such shell scripts, users can extend the capabilities of the LPS themselves. For details on invoking LPS functional programs from a shell, read Appendixes E and F.

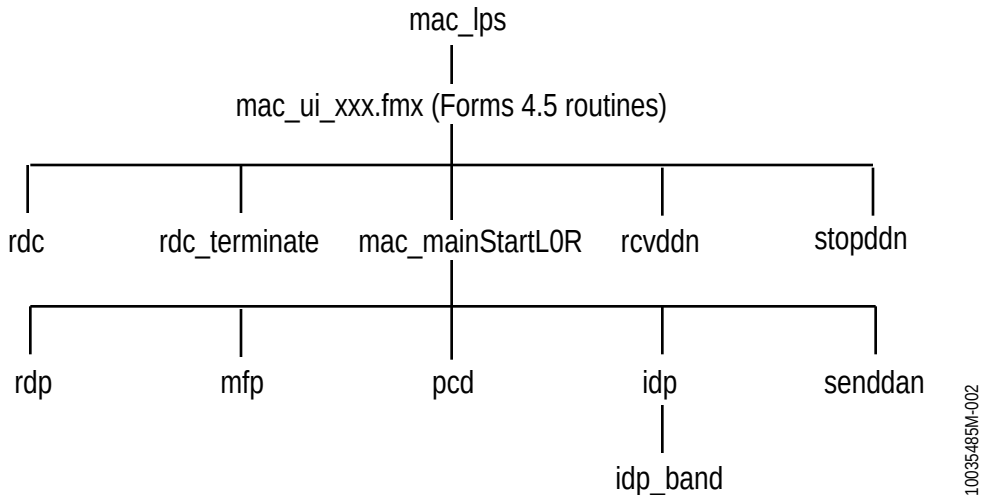

Figure 1–3. LPS Software Structure

1.3.5 Operator Role

The LPS operator plays an active role in LPS operations. The following subsections describe how the LPS operator uses LPS software to perform the basic LPS operations of data capture, L0R processing, and output file transfer, as well as operations for testing and maintenance.

1.3.5.1 Data Capture

The LPS operator's role in data capture is to provide contact schedules from the LGS for ingest into the databases on each LPS string; monitor automatic, scheduled data capture; and generate a Data Receive Summary Report after each contact and relay it to the Landsat 7 MOC. In more detail, the LPS operator's role is as follows:

- Use IRIX FTP (**TBR**) to retrieve contact schedules from the LGS.
- Use the "Ingest Contact Schedule..." command from the LPS GUI to ingest the contact schedule into each LPS string.
- Use the "Set Capture Source..." command from the LPS GUI to set the source of the incoming data each time it changes.
- Use the LPS "Data Receive Summary Report..." command from the LPS GUI to generate a Data Receive Summary Report after each contact for forwarding to the MOC.
- Monitor the capture of raw wideband data by reviewing the status and error messages generated by the LPS software.
- Ensure that a new tape is available for copying the raw wideband data to the 30-day store and monitor the copying operation by reviewing the status and error messages generated by the LPS software.

1.3.5.2 L0R Processing

The LPS operator's role in L0R processing is to monitor automatic L0R processing, verify L0R processing quality during and after the processing, and perform reprocessing when requested by the IAS. In more detail, the LPS operator's role is as follows:

- ¥ Monitor automatic L0R processing by reviewing the status and error messages generated by the LPS software.
- ¥ Set up the MWD to monitor L0R processing quality while the processing is in progress.
- ¥ Use the "LPS Q/A Report..." command to generate an LPS Q&A report to review L0R processing quality after processing has completed.
- ¥ Service reprocessing requests by using the "Start restage..." command to restage raw wideband data from tape and then use the "Start Level 0R Processing..." command to perform L0R processing.
- ¥ Restage data.

1.3.5.3 Output File Transfer

The LPS operator's role in output file transfer is to monitor the automatic transfer of files from the LPS to EDC DAAC. In more detail, the LPS operator's role is as follows:

- ¥ Monitor automatic file availability notification and file transfer disposition by reviewing the status and error messages generated by the LPS software.
- ¥ Review daily file transfer activity by using the "File Transfer Summary..." command from the LPS GUI to generate a File Transfer Summary Report.

1.3.5.4 Test and Maintenance

In addition to normal LPS operations, the LPS operator uses LPS software to test LPS functions. In more detail, the LPS operator's role is as follows:

- ¥ Test LPS data capture functions by using the "Send Test Data..." command on a source LPS string to play back a test data set and use the manual "Start Data Capture..." command on a destination LPS string to capture the data.
- ¥ Use the "DAN Transfer State..." command from the LPS GUI to disable automatic file availability notification and then use the "Start Level 0R Processing..." command from the LPS GUI to process a test data set to L0R.
- ¥ Monitor L0R quality using the MWD and LPS Q&A report as described in Section 1.3.5.2.

1.4 Notational Conventions

This users guide uses various type styles for different categories of terms. Table 1–1 describes these conventions.

Warnings and Notes appear throughout this users guide. Warnings alert the user to potentially destructive or hazardous actions. For example,

WARNING

Selecting “Commit” will overwrite the value in the database. The original value cannot be recovered.

Notes provide points of useful information. For example,

NOTE

When logging on to LPS strings from an X terminal, the desktop will be on the first LPS string to which you connected. Window manager menus will apply to that string.

Table 1–1. Text Style Conventions

Text Style	Description
Control>Start L0R ...	Menu items you select from the LPS GUI menu bar or other system menus are shown bold face in Arial font. The description shows the top-level menu name, the symbol ">", and the menu option. Pull-right menus may have additional entries. For example, **Setup>Parameters...>Raw Data Processing...** indicates the Raw Data Processing... menu option, which is in the Parameters... pull-right menu, which is itself under the Setup menu.
`% cat /etc/passwd \` `? > savefile`	Any text that you would type as an IRIX shell command appears in Courier font. "%" represents the IRIX command prompt and "?" represents the IRIX continuation prompt. Do not type % or ? when they appear in these positions. In this document, commands are sometimes split to accommodate the document's format. You can omit the continuation and type the entire command on a single line.

`SQL> SELECT *`
`2 FROM rdc_acct`
`3 WHERE csid = 123;`
	Any text that you would type as input to ORACLE SQL*Plus appears in Courier font. Structured Query Language (SQL) key words appear in upper case. Table and attribute names appear in lower case. "SQL" represents the SQL*Plus command prompt. "2" and "3" represent the SQL*Plus continuation prompt. Do not type them when they appear in these positions.
	
	In this document, commands are sometimes split to accommodate the document's format. You can omit the continuation and type the entire command on a single line.

`% telnet` *`String-Name`*	Italicized text indicates an item, such as a file name, that you must supply.
`% mac_startL0R [0 | 1]`	Items in square brackets separated by "|" indicate a set of options. Type one (and only one) of the options exactly as it appears.

2. Getting Started

2.1 Introduction

This chapter describes the fundamentals of operating the LPS software. It explains the steps required to

- Log on to the LPS.
- Set up an environment to run the LPS software.
- Start up and shut down the LPS software.
- Navigate the LPS GUI.
- Set up Status and Error Message displays.
- Browse the LPS Journal to view status and error history.

2.2 Logging On to the LPS

If you do not already have accounts on the LPS system hosts, contact the LPS system administrator. Be sure to tell the system administrator that your account must be an LPS operator account. The system administrator will provide your user name and initial password.

Once you have a user name and password, you can log on to the LPS. Normally, you will log on to each of the four LPS operational strings by logging onto two strings from each of the LPS X terminals. However, you can log onto any set of the strings from either of the X terminals or from the Indy workstations normally used for the MWD.

NOTE

When logging on to LPS strings from an X terminal, the desktop will be on the first LPS string to which you connected. Window manager menus will apply to that string.

2.2.1 Logging On From an X Terminal

To log on to the first LPS string from an X terminal:

1. If necessary, press the “Setup” key to display the X terminal menu.
2. If necessary, select the **Login>Login New X Session…** menu option to display the X session login dialog.
3. Make sure the **Net** field is set to **TCP** and the **Type** field is set to **Host**.
4.

Determine the Internet Protocol (IP) address of the X terminal by

- Selecting the **Setup>Change Setup Parameters…** menu option
- Clicking on the box to the left of **IP** to display the Internet Parameters Setup form
- Finding the Internet address of the X terminal in the "IP Address" field and making a note of it
- Closing the "Change the Setup Parameters..." menu by clicking the left mouse button over the bar in the upper left-hand corner of the window to pop up a menu and then clicking the left mouse button over the "Close" option
- Closing the "Setup" menu box by clicking the left mouse button over the bar in the upper left-hand corner of the window to pop up a menu and then clicking the left mouse button over the "Close" option

5. Connect to an LPS string by selecting its name from the **Default Hosts** list or by typing its name or IP address into the **Host field** (move the cursor into the typing area to enter the name) and then click **OK** with any mouse button.
6. A login dialog will appear after a few seconds. Type your user name and press return, then type your password (your password is not echoed on the screen) and press return.

A desktop on the LPS string will appear. To log on to additional LPS strings from an X terminal:

7. Select an existing xwsh or xterm window or create a new window.
8. In the selected window, type the following command:

   `% telnet` *`String-Name-or-IP-Address`*
9. At the `login:` prompt, type your user name and press return; at the `Password` prompt type your password (your password is not echoed on the screen) and press return.
10. Set the DISPLAY environment variable by typing

    `% setenv DISPLAY` *`X-Terminal-IP-Address`*`:0.0`

    `X-Terminal-IP-Address` is the IP address for the X terminal that you discovered in step 4.

Repeat steps 7 through 10 for each LPS string that you want to connect with from this X terminal.

2.2.2 Logging On From a Workstation

To log on to a LPS string from a workstation:

1. Log on to the workstation. Type your user name and press return, then type your password (your password is not echoed on the screen) and press return.
2. Select an existing window or create a new window.
3. In the selected window, type the following command:

```
% telnet String-Name
```

4. At the `login:` prompt, type your user name and press return; at the `Password` prompt, type your password (your password is not echoed on the screen) and press return.

Repeat steps 2 through 4 for each LPS string you want to connect with from this workstation.

2.3 Setting Up Your Environment

The LPS software includes a resource file that will set up your csh or tcsh environment properly [there is no support for Bourne (sh) shell or Korn (ksh) shell]. To invoke the file, add the following lines at the top of your .cshrc file on each LPS string:

```
% setenv LPS_HOME /user/LPS/st/r1 (this is an example)
% source $LPS_HOME/.lpsrc
```

NOTE

You have separate accounts on each LPS string and workstation and each uses a different .cshrc file. Lines similar to those above must appear in each of these files.

If necessary, you can modify any of the LPS elements. Appendix D describes how to do it.

2.4 User Permissions and Privileges

NOT IMPLEMENTED IN RELEASE 1

2.5 Starting Up the LPS Software

Once you have logged on to each LPS string, you will need to start up the LPS software on each string.

NOTE

A separate instance of the LPS software must be started on each LPS string.

To start the LPS software on a particular LPS string:

1. Move to the X terminal or workstation from which you are logged on to the string.
2. Use the mouse to move the cursor into a shell window for that string.
3.

Set up to display LPS startup messages in the window by typing:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk '{print $NF}'`
% tail -f $LPSJournal&
```

4. Start up the LPS software by typing

   ```
   % mac_lps
   ```

5. The LPS software will update raw data file accounting and LPS process information in the LPS database. Any error messages encountered will both display in your window and be written to the LPS Journal file.

6. If the database is updated successfully, the LPS GUI will start up.

7. Stop the display of LPS messages to the window by typing `% killall tail` in the window.

2.6 Navigating the LPS GUI

You control the LPS through the LPS software GUI. The LPS GUI displays as a menu bar (Figure 2–1).

Figure 2–1. LPS GUI Menu Bar

NOTE

Each LPS string has a separate GUI. Commands issued through the GUI apply only to that string. The GUI displays the name of the string on which it is executing.

To issue a command through the LPS GUI on a particular LPS string:

1. Hold down the left mouse button over the desired menu item.
2. While still holding down the mouse button, move the cursor to highlight the selection you want to invoke.
3. Release the mouse button.

In most cases, a dialog form will appear asking you to confirm your selection and possibly asking for further information. Details for each command accessible through the LPS GUI and instructions for providing any further information appear in Section 10.

2.7 Status and Error Message Displays

All of the LPS software writes status and error messages to an LPS Journal file. You monitor LPS operations by reviewing the messages sent to the LPS Journal. To view messages as they are sent to the LPS Journal, set up a status/error message display window after you start up the LPS software on each string. You can also browse the LPS Journal file to review LPS processing history.

NOTE

Each string has a separate LPS Journal file. Status and error messages appear only in the Journal file for the string on which they were generated.

2.7.1 Setting Up Message Displays

To set up a status/error message display:

1. Select the **MONITOR>Display Operations Messages...** menu option. The LPS GUI will display a dialog allowing you to select the priorities of the messages to be displayed.
2. If necessary, select priorities for display. The LPS software will display only messages with priorities that you have selected. The default is to display messages of all priorities except DEBUG.
3. Click the OK button; the LPS software will display a new window with 10 messages already written to the LPS Journal file.

NOTE

Any number of message displays can be created displaying messages of different priorities. For example, you could create one window displaying only high-priority error messages and another displaying informational messages. Because fewer messages go to the high-priority message window, it is less likely that an important message will scroll off the screen before you can see it.

The following is an example of a line from the LPS Journal file as displayed in the “Display Operations Messages...” window:

```
15:36:22 INFO syslog: 3475 [ldt_establish_client.c:270] ECS
accepted authentication request.
```

The time of the message appears first, followed by its priority (INFO in the example) and source. Table 2–1 summarizes the priorities and their meanings. For messages generated by the LPS software, the source is always "syslog." The process group identifier (ID) of the process generating the message appears next (3475 in the example). The unit name and source line appear next in square brackets. These are normally of interest only for software troubleshooting. The final part of the line is the message itself. Note that the message wraps to fit within the window.

Table 2–1. LPS Message Priorities

Priority No.	Description
EMERG	EMERGENCY – system is unusable
ALERT	ALERT – immediate action must be taken
CRIT	CRITICAL – critical condition
ERROR	ERROR – error condition
WARN	WARNING – warning condition
NOTICE	NOTICE – normal but significant
INFO	INFO – informational message
DEBUG	DEBUG – debug level messages intended for software trouble-shooting, not of operational interest

Normally, you will create a status/error message display window for each LPS string by selecting menu options from the LPS GUI. It is also possible to view LPS messages as they are written to the LPS Journal file from an IRIX shell.

2.7.2 Browsing the LPS Journal

To browse the LPS Journal:

1. Select the **MONITOR>Display LPS Journal File...** menu option. The LPS GUI will bring up the IRIX sysmon(1) utility to display the LPS Journal file (Figure 2–2).
2. If desired, customize the display by selecting options from the sysmon(1) menu bar. Consult the online documentation for sysmon(1) for help in customizing the display by clicking the "Help" option on the far right of the sysmon(1) window menu bar.

Figure 2–2. LPS Journal File Displayed by sysmon(1)

2.8 Shutting Down the LPS Software

To shut down the LPS software on a particular string:

1. Verify that the DDN handler (rcvddn) is not currently processing a DDN. For Release 1, determine whether the DDN handler is active by reviewing the LPS Journal file for DDN-related messages.

   WARNING

   For Release 1, stopping the DDN server while it is processing a DDN will cause it to terminate prematurely, possibly leaving the LPS in an inconsistent state.

2. Select the **Shutdown** menu option.

3. If the LPS software displays a dialog form stating that LPS processes are active and you still wish to shut down the LPS software:

   3a. If the **Stop Data Capture...** option is displayed under the **Control** menu, select it and click the "OK" button when the confirmation dialog appears.

3b. If the **Stop Data Processing...** option is displayed under the **Control** menu, select it. When the Stop Data Processing... dialog appears, select an instance of L0R processing from the menu and click the OK button. Repeat selecting **Control >Stop Data Processing...** and selecting another L0R processing instance until the **Stop Data Processing...** option no longer appears under the **Control** menu.

3c. Select the **Shutdown** menu option.

4. Click the OK button in the confirmation dialog. Clicking the CANCEL button will leave the LPS software up and running.

2.9 Logging Off the LPS

Logging off from the LPS strings in an orderly manner ensures that all resources are freed and that all functions to be executed on exit are performed.

To log off from each LPS string to which you are connected by telnet:

1. Exit from all programs executing in separate windows, such as LPS Journal file displays using xwsh, windows-based editors, and sysmon.
2. Verify that the LPS GUI has been shut down and that no LPS software processes invoked from the command line are still running (see Section 2.8).
3. Either type `logout` in the telnet session window or select **Quit** from the window's right mouse button menu.

To log off from an LPS string to which you are directly logged on:

4. Select **Logout** from the desktop's right mouse button menu.
5. Click the **Yes** button in the confirmation dialog.
6. If you are logged on through an X terminal, click the **Yes** button in the X terminal's confirmation dialog.

3. Data Capture Operations

3.1 Introduction

This chapter describes how to perform operations related to LPS data capture. It describes how to

- Manage LPS automatic data capture operations.
- Capture data manually.
- Monitor data capture.

3.2 Managing Automatic Data Capture

This section describes how to perform operations related to LPS automatic data capture. It describes how to

- Fetch a contact schedule from the LGS.
- View the LGS contact schedule file.
- Ingest the contact schedule file into a string database.
- View or edit the contact schedule stored in a string database.
- View or edit a string's capture source.
- Verify whether automatic data capture is active.

3.2.1 Fetching a Contact Schedule From the LGS

NOT IMPLEMENTED IN RELEASE 1

3.2.2 Viewing the LGS Contact Schedule File

NOT IMPLEMENTED IN RELEASE 1

3.2.3 Ingesting a Contact Schedule File

NOT IMPLEMENTED IN RELEASE 1

3.2.4 Viewing or Editing a String's Contact Schedule

NOT IMPLEMENTED IN RELEASE 1

3.2.5 Viewing a String's Capture Source

NOTE

For Release 1, access to the capture source is possible only through SQL*Plus.

To determine the value of a string's capture source, type the following to SQL*Plus:

```
SQL> SELECT capture_source(TBR) FROM lps_configuration;
```

The value returned is a two-character encoding of the possible sources of data. See Table 10–1 for explanations of their meanings.

3.2.6 Editing a String's Capture Source

NOTE

For Release 1, access to the capture source is possible only through SQL*Plus.

WARNING

The LPS software uses the capture source to determine which entries in the contact schedule apply to the string. An incorrect capture source value may result in the failure to capture data on schedule.

The LPS software also uses the capture source to identify the data in its accounting. An incorrect capture source value may result in misidentification of captured data.

To set the value of a string's capture source, type the following to SQL*Plus:

```
SQL> UPDATE lps_configuration SET capture_source(TBR) =
'Value' ;
```

Value is a two-character encoding of the possible sources of data. See Table 10–1 for explanations of their meanings.

3.2.7 Verifying That Automatic Data Capture Is Active

NOT IMPLEMENTED IN RELEASE 1

3.3 Manual Data Capture

Normally, the LPS software captures data automatically according to the contact schedule you receive from the LGS. For testing or for handling emergency situations, however, you may need to start and stop data capture yourself.

WARNING

Data capture is independent on each LPS string. To capture all data channels, you must command each LPS string to capture data.

You will normally capture data manually through the LPS GUI. It is possible, though less convenient, to do so directly from an IRIX shell. The shell interface is intended for emergency situations in which capture must occur when the LPS GUI is not up and running and for use in shell scripts you may create to simplify manual operations. For details see Appendix E, Section E.3.1.

3.3.1 Starting Data Capture

To start data capture manually on an LPS string:

1. Select **Control>Start Data Capture...** from the LPS GUI. This menu option is available only if the LPS software is not currently capturing raw wideband data.
2. The LPS software displays a Start Capture dialog (Figure 10–4).
3. If necessary, set the capture source to the source of the data to be captured by holding down the mouse button in the capture source field and selecting the appropriate value from the pop-up menu. See Table 10–1 for a description of the capture source values.
4. If necessary, set the capture stop time to the time at which you want capture to stop. The format for the time you input should be YYYY:DDD:HH:MM:SS. The LPS software will verify that this time is in the correct format and is later than the current time. It will issue an error message if the time is incorrect. Data capture will not start if the time you enter fails either of these checks. The default time is 14 minutes from the time you selected the **Control>Start Data Capture...** command.
5. If necessary, set the capture process execution characteristics controls (not implemented in Release 1).
6. Click the OK button.
7. Click the **Yes** button in the confirmation dialog.

Data capture begins as soon as you click the **Yes** button, though there is a small delay as the LPS software sets up to begin capture. Capture continues until either the stop time you specified is reached, a maximum duration of 14 minutes elapses, or you manually stop data capture.

3.3.2 Stopping Data Capture

To stop data capture manually on an LPS string:

1. Select **Control>Stop Data Capture...** from the LPS GUI. This menu option is available only if the LPS software is currently capturing raw wideband data.
2. The LPS software displays a Stop Capture dialog (Figure 10–5) asking you to confirm your command.

3. Click the OK button.

Data capture stops as soon as you click the OK button.

3.4 Monitoring Data Capture

To monitor data capture operations, view the status and error messages generated by the mac_AutoCapture and rdc processes. To view status and error messages generated by these processes, set up a real-time display of the LPS Journal file (see Section 2.7.2).

4. Raw Wideband Data Management

NOT IMPLEMENTED IN RELEASE 1

5. Level 0R Processing Management

5.1 Introduction

This chapter describes how to perform operations related to LPS L0R processing. It describes how to

- Manage automatic L0R processing operations.
- Perform L0R processing manually through the LPS GUI.
- Monitor L0R processing.
- View and edit L0R processing parameters.
- View and edit L0R processing error reporting thresholds.

5.2 Automatic L0R Processing Management

NOT IMPLEMENTED IN RELEASE 1

5.3 Manual L0R Processing

This section describes how to manually control LPS L0R processing. You must start L0R processing manually under the following conditions:

- To process a raw data file for which automatic L0R processing failed to complete
- To handle a reprocessing request
- To perform tests

You can stop any L0R processing, whether it was started automatically or manually.

You will normally control L0R processing manually through the LPS GUI. It is also possible to control L0R processing directly from an IRIX shell (see Appendix E). The IRIX interface is intended for use in shell scripts you may create.

NOTE

When you stop L0R processing manually, all output files created by the processing will be deleted.

5.3.1 Starting L0R Processing

To start L0R processing manually:

1. Select the **Control>Start Level 0R Processing...** command from the LPS GUI. The "Start Level 0R Processing" dialog appears.
2. Select the raw wideband data file you want to process by clicking on any of its information fields.
3. If desired, change the settings for the options to delete the raw data file after successful processing (not selected by default) or to delete the output files automatically when transferred (selected by default).
4. Click the OK button. A confirmation dialog appears.
5. Click the OK button.

5.4 Monitoring L0R Processing

There are two ways to monitor L0R processing:

1. *Viewing L0R messages* – A status message indicates that L0R processing has completed. L0R error messages indicate when the amount of erroneous data encountered exceeds the user-specified threshold. Error messages also indicate (generally fatal) processing errors encountered by the LPS L0R processing software.
2. *Using the MWD* – The MWD (not implemented in Release 1) provides a color thumbnail image of the band data as it is being processed. The image gives a rough idea of processing quality.

5.4.1 Viewing L0R Status and Error Messages

To view status and error messages generated by the LPS L0R processing software, set up a real-time display of the LPS Journal file by any of the methods described in Section 2.7.2. The LPS software writes all status and error messages to the LPS Journal file (and to the standard output, if the LPS_LOG_STDOUT environment variable is set).

5.4.2 Moving Window Display

NOT IMPLEMENTED IN RELEASE 1

5.5 L0R Processing Parameters Management

This section describes how to view and modify the parameters that the LPS software uses during L0R processing. L0R processing parameters are values that control certain aspects of L0R processing, such as the number of bit errors to allow in a frame synchronization marker or the maximum forward time jump length allowed within a subinterval. The L0R processing parameters fall into the following categories:

- *Raw Data Processing* (**TBR**) – Parameters that control Consultative Committee on Space Data Systems (CCSDS) frame synchronization and error detection and correction.
- *Major Frame Processing* (**TBR**) – Parameters that control the reconstruction of ETM+ telemetry data and the detection of subintervals within the contact.
- *PCD Processing* (**TBR**) – Parameters that control the reconstruction of PCD telemetry data and the identification of WRS scenes.
- *Band Processing* (**TBR**) – Parameters that control browse file generation and define the Automated Cloud Cover Assessment (ACCA) method string in LPS metadata files.

NOTE

For Release 1, access to L0R processing parameters is possible only through SQL*Plus.

Not all L0R processing parameters are implemented for Release 1.

5.5.1 Viewing L0R Processing Parameters

This subsection describes how to view the current values of the L0R processing parameters on a LPS string.

NOTE

Each LPS string has its own set of parameters. Only the LPS string's own set of parameters may be viewed from an IRIX shell on that string.

To view the current value of any set of L0R processing parameter from ORACLE SQL*Plus:

1. Find the table and attribute names for each L0R processing parameter you want to view in Table 5–1.
2. In SQL*Plus, type the following:

```
SQL> SELECT Attributes FROM Tables
2    [ WHERE insertion_time IN
3      (SELECT MAX(insertion_time) FROM valid_ccsds_parms) ] ;
```

Attributes is a comma-separated list of the attribute names found in step 1. *Tables* is a comma-separated list of the table names found in step 1. The WHERE clause is required if, and only if, "valid_ccsds_parms" is in *Tables*.

Table 5–1. Table and Attribute Names for L0R Processing Parameters (1 of 5)

Parameter	Table Name	Attribute Name(s)
ACCA		

CCA Method String appearing as CCA method in metadata files	valid_band_parms	cca_method
ACCA Subsampling Ratio ACCA reduction ratio	valid_band_parms	CCA_ratio
Browse		
Browse Bands The three bands to include in the browse image NOTE: Bands must be included in format 1.	valid_band_parms	multi1, multi2, multi3
Browse Subsampling Ratio Browse image subsampling reduction ratio	valid_band_parms	subs
Browse Wavelets Iterations Number of times to apply wavelets reduction during browse generation	valid_band_parms	wave
Radiometric Calibration Gains Gains used for ACCA and browse image radiometric correction; there are separate values for high and low gains for each detector in each band for a total of 272 values	Not in Release 1	Not in Release 1
Radiometric Calibration Biases Biases used for ACCA and browse image radiometric correction; there are separate values for high and low gains for each detector in each band for a total of 272 values	Not in Release 1	Not in Release 1
CCSDS Frame Synchronization		
Bit Slip Correction Extent Number of bits around the expected frame sync pattern start bit to search for the pattern	valid_ccsds_parms	cadu_bit_slip_ correction_extent

Table 5–1. Table and Attribute Names for L0R Processing Parameters (2 of 5)

Parameter	Table Name	Attribute Name(s)
Check Sync Marker Error Tolerance Number of bit errors in the frame synchronization pattern to accept and remain in Check mode	valid_ccsds_parms	cadu_sync_mark_ check_error_tol
Check Tolerance Number of channel access data unit (CADU) frame synchronization errors to tolerate while in Check mode before going back to Search mode	valid_ccsds_parms	cadu_check_tol
Flywheel Tolerance Number of CADU frame synchronization errors to tolerate while in lock mode before going back to Search mode	valid_ccsds_parms	cadu_flywheel_tol
Lock Sync Marker Error Tolerance Number of bit errors in the frame synchronization pattern to accept and remain in Lock mode	valid_ccsds_parms	cadu_sync_lock_ error_tol
Search Tolerance Number of successful CADU frame synchronizations required before moving from Search to Check mode	valid_ccsds_parms	cadu_search_tol
ETM+ Processing		
ETM_ Fill Pattern Two-byte pattern to use as fill for missing ETM+ telemetry	valid_mfp_parms	fill_value
ETM+ Majority Vote Tolerance Tolerable number of differing bit values in any 40-bit majority vote	valid_mfp_parms	maj_vote_tol
Maximum Alignment Value Total number of scan line alignment pixels	valid_mfp_parms	max_alignment_ value
Nominal ETM+ Rate Nominal major frame transmission rate in **TBD** units	valid_mfp_parms	mjf_data_period

Table 5–1. Table and Attribute Names for L0R Processing Parameters (3 of 5)

Parameter	Table Name	Attribute Name(s)
Sensor Alignments Adjustment in Instantaneous Field-of-View (IFOV)s to layout geometry and multiplexer sampling; there are separate values for forward and reverse scans for odd and even detectors for each band for a total of 32 values (forward odd, forward even, reverse odd, and reverse even for Bands 1 through 8)	valid_mfp_parms	sensor_alignment_***A_B***_Band***C_D*** ***A*** = 'forward' or 'reverse' ***B*** = 'east' or 'west' ***C*** = 1..8 ***D*** = 'even' or 'odd'
Subinterval Delta Time Maximum time gap between ETM+ major frame times that will be filled; time gaps greater than this value will cause telemetry to be placed in a new subinterval	valid_mfp_parms	sub_intv_delta
Time Range Tolerance Tolerable difference between the actual ETM+ major frame timecode value and the expected value	valid_mfp_parms	time_range_tol
LPS Configuration Metadata		
IAS Parameter File Version Number Version number of the IAS parameter file used by the LPS string	lps_configuration	IAS_PARAM_FILE_VER_NUM
LPS Software Version Number Version number of the LPS software installed on the LPS string	lps_configuration	lps_software_version_number
LPS String ID Character string used to identify the LPS string	lps_configuration	lps_hardware_string_id
Spacecraft Identifier (SCID) Character string used as SCID in metadata files	lps_configuration	spacecraft_id

Table 5–1. Table and Attribute Names for L0R Processing Parameters (4 of 5)

Parameter	Table Name	Attribute Name(s)
PCD Processing		
PCD Fill Pattern Byte value used as fill for missing PCD telemetry	valid_pcd_parms	pcd_frame_fill
WRS Scene Identification		
Attitude Quaternion Tolerance Maximum tolerable absolute value of the difference between the sum of squares of the attitude quaternion and 1.0	valid_pcd_thres	attitude_ quaternion_tol
Earth Semi-Major Axis The Earth's semi-major axis in **TBD** units	valid_scene_parms	semi_major_axis
Earth Semi-Minor Axis The Earth's semi-minor axis in **TBD** units	valid_scene_parms	semi_minor_axis
Ephemeris Position Lower Bound Lower bound on valid ephemeris position vector magnitude	valid_pcd_thres	ephem_position_ lower
Ephemeris Position Upper Bound Upper bound on valid ephemeris position	valid_pcd_thres	ephem_position_ upper
Ephemeris Velocity Lower Bound Lower bound on valid ephemeris velocity vector magnitude	valid_pcd_thres	ephem_velocity_ lower
Ephemeris Velocity Upper Bound Upper bound on valid ephemeris velocity vector magnitude	valid_pcd_thres	ephem_velocity_ upper

Table 5–1. Table and Attribute Names for L0R Processing Parameters (5 of 5)

Parameter	Table Name	Attribute Name(s)
ETM+ to Attitude Control Reference Axis Matrix A matrix describing the relationship of the ETM+ optical axis to the satellite's attitude control reference axis; each element of the matrix is stored as a separate field in the LPS database	valid_scene_parms	etm_body_ transmatrix_***XY*** ***X*** = 1..3 ***Y*** = 1..3
UT1 to UTC Time Conversion Coefficients X shift pole wander in arc seconds, Y shift pole wander in arc seconds, and UT1-UTC time difference to be used for UT1 to UTC time conversions for each day of a 180-day period. Each value is stored as a separate field in the database. Each set of three values for each day is tagged with the day on which it is valid.	**TBD**	**TBD**

Example: To view the values of the frame synchronization search tolerance and the value of the ETM+ processing subinterval delta time:

1. From Table 5–1, the search tolerance has the table name "valid_ccsds_parms" and the attribute name "cadu_search_tol." The subinterval delta time has the table name "valid_mfp_parms" and the attribute name "sub_intv_delta."
2. In SQL*Plus, type the following:

```
SQL> SELECT cadu_search_tol, sub_intv_delta
2 FROM valid_ccsds_parms, valid_mfp_parms
2    WHERE insertion_time IN
3      (SELECT MAX(insertion_time) FROM valid_ccsds_parms);
```

5.5.2 Ingesting IAS Parameter Files

NOT IMPLEMENTED IN RELEASE 1

5.5.3 Updating IAS Parameters Manually

To update the value of any IAS parameter from ORACLE SQL*Plus:

1. Find the table and attribute name for the parameter you want to update in Table 5–1.
2. Verify that the parameter's new value is within its valid range as listed in Table 5–2.

Table 5–2. IAS L0R Parameters Suggested Values and Valid Ranges

Parameter	Suggested Value	Valid Range	Description
Browse			
Radiometric Calibration Gains	Not in Release 1	Not in Release 1	Gains used for browse image radiometric correction; there are separate values for high and low gains for each detector in each band for a total of 272 values
Radiometric Calibration Biases	Not in Release 1	Not in Release 1	Biases used for browse image radiometric correction; there are separate values for high and low gains for each detector in each band for a total of 272 values
ETM+ Processing			
Sensor Alignments	**TBD**	0–99999	Adjustment in IFOVs to layout geometry and multiplexer sampling; there are separate values for forward and reverse scans for odd and even detectors for each band for a total of 32 values (forward odd, forward even, reverse odd, and reverse even for Bands 1 through 8)
LPS Configuration Metadata			
IAS Parameter File Version Number	N/A	1–9999	Version number of IAS parameter file used by the LPS string
WRS Scene Identification			
Earth Semi-Major Axis	**TBA**	**TBA**	The Earth's semi-major axis in **TBD** units
Earth Semi-Minor Axis	**TBA**	**TBA**	The Earth' semi-minor axis in **TBD** units
ETM+ to Attitude Control Reference Axis Matrix	**TBA**	-1.00–+1.00	A matrix describing the relationship of the ETM+ optical axis to the satellite's attitude control reference axis; each element of the matrix is stored as a separate field in the LPS database
UT1 to UTC Time Coefficients	**TBD**	**TBD**	UT1-UTC time difference to be used for UT1 to UTC time conversions for each day of a 180-day period; each value is stored as a separate row in the database tagged with the day on which it is valid

3. In SQL*Plus, type the following:

   ```
   SQL> UPDATE Table SET Attributes = Value;
   ```

   Table is the table name found in step 1. *Attribute* is the attribute name found in step 1. *Value* is the new value to set the parameter to.

4. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```

5. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   ```

6. Verify that the changes have been saved by noting that the following message appears:

   ```
   Commit complete.
   ```

Example: To set the value of the ephemeris position upper bound to **TBD**:

1. From Table 5–1, the ephemeris position upper bound has the table name "valid_pcd_thres" and the attribute name "ephem_position_upper."
2. From Table 5–2, **TBD** is within the valid range for ephemeris upper bound.
3. In SQL*Plus, type the following:

   ```
   SQL> UPDATE valid_pcd_thres SET ephem_position_upper = 3;
   1 row updated.
   ```

4. The message "1 row updated." indicates the update has occurred successfully.
5. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   Commit complete.
   ```

6. The "Commit complete." message indicates that the changes have been saved.

5.5.4 Updating Other L0R Parameters Manually

To update the value of non-IAS parameters from ORACLE SQL*Plus:

1. Find the table and attribute name for the parameter you want to update in Table 5–1.
2. Verify that the parameter's new value is within its valid range as listed in Table 5–3.

Table 5–3. Non-IAS L0R Parameters Suggested Values and Valid Ranges (1 of 3)

Parameter	Suggested Value	Valid Range	Description
ACCA			
CCA Method	**TBD**	Not null character string	String appearing as CCA method in metadata files
ACCA Subsampling Ratio	**TBA**	4, 8, 16, 32, 48	ACCA reduction ratio
Browse			
Browse Bands	**TBD**	Three of 1, 2, 3, 4, 5, 6	The three bands to include in the browse image NOTE: Bands must be included in format 1.
Browse Subsampling Ratio	Not in Release 1	Not in Release 1	Browse image subsampling reduction ratio
Browse Wavelets Iterations	Not in Release 1	Not in Release 1	Number of times to apply wavelets reduction during browse generation
CCSDS Frame Synchronization			
Bit Slip Correction Extent	0	0…3	Number of bits around the expected frame sync pattern start bit to search for the pattern
Check Sync Marker Error Tolerance	0	0…3	Number of bit errors in the frame sync pattern to accept and remain in Check mode
Check Tolerance	0	0…3	Number of CADU frame sync errors to tolerate while in Check mode before going back to Search mode
Flywheel Tolerance	0	0…3	Number of CADU frame sync errors to tolerate while in lock mode before going back to Search mode
Lock Sync Marker Error Tolerance	0	0…3	Number of bit errors in the frame sync patter to accept and remain in Lock mode

Table 5–3. Non-IAS L0R Parameters Suggested Values and Valid Ranges (2 of 3)

Parameter	Suggested Value	Valid Range	Description
Search Tolerance	1	1…3	Number of successful CADU frame synchronizations required before moving from Search to Check mode
ETM+ Processing			
ETM+ Fill Pattern	**TBD**	**TBD**	Byte pattern to use as fill for missing ETM+ telemetry
ETM+ Majority Vote Tolerance	**TBA**	**TBA**	Tolerable number of differing bit values in any 40-bit majority vote
Maximum Alignment Value	**TBA**	**TBA**	Total number of scan line alignment pixels
Nominal ETM+ Rate	**TBA**	**TBA**	Nominal major frame transmission rate in **TBD** units
Subinterval Delta Time	**TBA**	**TBA**	Maximum time gap between ETM+ major frame times that will be filled; time gaps greater than this value will cause telemetry to be placed in a new subinterval
Time Range Tolerance	**TBD**	0.00–9999999.99	Tolerable difference between the actual ETM+ major frame timecode value and the expected value
LPS Configuration Metadata			
Instrument ID	ETM+	Character string	Character string used as instrument ID in metadata files
LPS Software Version Number	Not applicable	Character string	Version number of the LPS software installed on the LPS string
LPS String ID	Must be identical to IRIX host name	Must be identical to IRIX host name	Character string used to identify the LPS string in **TBD** cases
SCID	Landsat 7	Character string	Character string used as SCID in metadata files

Table 5–3. Non-IAS L0R Parameters Suggested Values and Valid Ranges (3 of 3)

Parameter	Suggested Value	Valid Range	Description
PCD Processing			
PCD Fill Pattern	**TBD**	**TBD**	Byte value used as fill for missing PCD telemetry
WRS Scene Identification			
Attitude Quaternion Tolerance	**TBD**	0.00–9999.99	Maximum tolerable absolute value of the difference between the sum of squares of the attitude quaternion and 1.0
Ephemeris Position Lower Bound	**TBD**	$-8.3886 \times 10^6 \leq x \leq -8.3886 \times 10^6$	Lower bound on valid ephemeris position vector magnitude (meters)
Ephemeris Position Upper Bound	**TBD**	$-8.3886 \times 10^6 \leq x \leq -8.3886 \times 10^6$	Upper bound on valid ephemeris position vector magnitude (meters)
Ephemeris Velocity Lower Bound	**TBD**	-8.0 m/ms $\leq x \leq$ -8.0 m/ms	Lower bound on valid ephemeris velocity vector magnitude (meters per millisecond)
Ephemeris Velocity Upper Bound	**TBD**	-8.0 m/ms $\leq x \leq$ -8.0 m/ms	Upper bound on valid ephemeris velocity vector magnitude (meters per millisecond)

3. In SQL*Plus, type the following:

```
SQL> UPDATE Table SET Attributes = Value
2    [ WHERE insertion_time IN
3      (SELECT MAX(insertion_time) FROM valid_ccsds_parms) ] ;
```

Table is the table name found in step 1. *Attribute* is the attribute name found in step 1. *Value* is the new value to set the parameter to. The WHERE clause is required if, and only if, the *Table* is "valid_ccsds_parms."

4. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

5. Save the change by typing the following to SQL*Plus:

```
SQL> COMMIT;
```

6. Verify that the changes have been saved by noting that the following message appears:

```
Commit complete.
```

Example: To set the value of the CCSDS frame synchronization search tolerance to 3:

1. From Table 5–1, the search tolerance has the table name "valid_ccsds_parms" and the attribute name "cadu_search_tol."
2. From Table 5–3, "3" is within the valid range for CCSDS frame synchronization search tolerance.
3. In SQL*Plus, type the following:

```
SQL> UPDATE valid_ccsds_parms SET cadu_search_tol = 3
2    WHERE insertion_time IN
3      (SELECT MAX(insertion_time) FROM valid_ccsds_parms);
1 row updated.
```

4. The message "1 row updated." indicates the update has occurred successfully.
5. Save the change by typing the following to SQL*Plus:

```
SQL> COMMIT;
Commit complete.
```

6. The "Commit complete." message indicates that the changes have been saved.

5.5.5 Propagating Parameters to Other Strings

NOT IMPLEMENTED IN RELEASE 1

5.5.6 Checking for Parameter Consistency Across LPS Strings

NOT IMPLEMENTED IN RELEASE 1

5.6 L0R Error Reporting Management

When the raw wideband data being processed contains many errors, the LPS L0R processing software can generate a large number of messages reporting that these errors have been encountered. The large number of messages can make it difficult to monitor LPS operations. Users can reduce the number of errors the LPS reports during L0R processing by setting any of a large number of error reporting thresholds.

When an error's threshold is set to a number greater than zero, the LPS L0R processing software will report errors only when the total number of errors in the processing run exceeds an integer multiple of the threshold. For example, if the threshold for CADU frame synchronization errors

is set to 100, the LPS L0R processing software will report CADU frame synchronization errors when the total number of such errors exceeds 100, again when it exceeds 200, and so on.

This section describes how to view and edit the thresholds for LPS L0R error reporting.

NOTE

For Release 1, access to L0R error reporting thresholds is possible only through SQL*Plus.

Not all L0R thresholds are implemented for Release 1.

5.6.1 Viewing L0R Processing Thresholds

To view the current value of any set of L0R error reporting thresholds from ORACLE SQL*Plus:

1. Find the table and attribute names for each L0R error reporting threshold you want to view in Table 5–4.
2. In SQL*Plus, type the following:

   ```
   SQL> SELECT Attributes FROM Tables ;
   ```

   Attributes is a comma-separated list of the attribute names found in step 1. *Tables* is a comma-separated list of the table names found in step 1.

Example: To view the values for the Bose-Chaudhuri-Hocquenghem (BCH) errors and cyclic redundancy code (CRC) errors thresholds:

1. From Table 5–4, the BCH errors threshold has the table name “valid_rdp_thres” and the attribute name “bch_thres.” The CRC errors threshold has the table name “valid_rdp_thres” and the attribute name “crc_thres.”
2. In SQL*Plus, type the following:

   ```
   SQL> SELECT bch_thres, crc_thres FROM valid_rdp_thres;
   ```

5.6.1.1 From the LPS GUI

NOT IMPLEMENTED IN RELEASE 1

5.6.2 Editing L0R Processing Thresholds

To modify the value of L0R error reporting thresholds from ORACLE SQL*Plus:

1. Find the table and attribute name for the parameter you want to update in Table 5–4.

Table 5–4. Table and Attribute Names for L0R Error Thresholds (1 of 2)

Threshold	Table Name	Attribute Name
Transfer Frame Processing		

BCH Errors Number of frames with uncorrectable errors in either the mission data or data pointer zones detected by BCH decoding	valid_rdp_thres	bch_thres
CADU Frame Sync Errors Sum of frames with sync pattern errors and flywheel frames detected	valid_rdp_thres	sync_thres
CRC Errors Number of frames with errors detected by CRC decoding	valid_rdp_thres	crc_thres
Reed-Solomon Errors Number of frames with uncorrectable errors detected by Reed-Solomon decoding	valid_rdp_thres	rs_thres
ETM+ Processing		
CADU Sequence Errors Modulo number of ETM+ sequence counter errors allowed before operator notification	valid_mfp_thres	mjf_cadu_seq_err_thres
End-of-Line Code Errors Modulo number of ETM+ end-of-line errors allowed before operator notification	valid_mfp_thres	eol_thres
Fully Filled Major Frames Modulo number of entirely filled ETM+ major frames allowed before operator notification	valid_mfp_thres	full_mjf_thres
Major Frame Sync Errors Modulo number of ETM+ major frame synchronization errors allowed before operator notification	valid_mfp_thres	mjf_sync_thres
Minor Frame Errors Modulo number of ETM+ minor frame errors allowed before operator notification	valid_mfp_thres	mnf_ctr_thres

Table 5–4. Table and Attribute Names for L0R Error Thresholds (2 of 2)

Threshold	Table Name	Attribute Name
Scan Direction Errors Modulo number of scan direction errors allowed before operator notification	valid_mfp_thres	scan_dir_thres
Time Code Errors Modulo number of ETM+ time code errors allowed before operator notification	valid_mfp_thres	tc_thres
Partially Filled Major Frames Modulo number of partially filled ETM+ major frames allowed before operator notification	valid_mfp_thres	part_mjf_thres
PCD Processing		
Majority Vote Failures Modulo number of failed PCD majority votes allowed before operator notification	valid_pcd_thres	num_failed_votes
Missing Data Words Modulo number of missing PCD words allowed before operator notification	valid_pcd_thres	num_missing_data_words

2. Verify that the threshold's new value is within its valid range as listed in Table 5–5.
3. In SQL*Plus, type the following:

   ```
   SQL> UPDATE Table SET Attribute = Value
   ```

   Table is the table name found in step 1. *Attribute* is the attribute name found in step 1. *Value* is the new value to set the parameter to.
4. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```
5. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   ```

Table 5–5. Error Reporting Thresholds Suggested Values and Valid Ranges

Threshold	Suggested Value	Valid Range	Description
Transfer Frame Processing			
CADU Frame Sync	0	0...2147483648	Sum of frames with sync pattern errors and flywheel frames detected
CRC	0	0...2147483648	Number of frames with errors detected by CRC decoding
BCH	0	0...2147483648	Number of frames with uncorrectable errors in either the mission data or data pointer zones detected by BCH decoding
Reed-Solomon	0	0...2147483648	Number of frames with uncorrectable errors detected by Reed-Solomon decoding
ETM+ Processing			
CADU Sequence Errors	**TBD**	0–99999999	Modulo number of ETM+ sequence counter errors allowed before operator notification
Scan Direction Errors	**TBD**	0–99999999	Modulo number of scan direction errors allowed before operator notification
Major Frame Sync Errors	**TBD**	0–99999999	Modulo number of ETM+ major frame synchronization errors allowed before operator notification
End-of-Line Code Errors	**TBD**	0–99999999	Modulo number of ETM+ end-of-line errors allowed before operator notification
Time Code Errors	**TBD**	0–99999999	Modulo number of ETM+ time code errors allowed before operator notification
Fully Filled Major Frames	**TBD**	0–99999999	Modulo number of entirely filled ETM+ major frames allowed before operator notification
PCD Processing			
Majority Vote Failures	**TBD**	0–99999999	Modulo number of failed PCD majority votes allowed before operator notification
Missing Data Words	**TBD**	0–99999999	Modulo number of missing PCD words allowed before operator notification

6. Verify that the changes have been saved by noting that the following message appears:

```
Commit complete.
```

Example: To set the value of the CRC threshold to 200:

1. From Table 5–4, the CRC threshold has the table name “valid_rdp_thres” and the attribute name “crc_thres.”
2. From Table 5–5, “200” is within the valid range for the CRC threshold (0....2147483648).
3. In SQL*Plus, type the following:

   ```
   SQL> UPDATE valid_rpd_thres SET crc_thres = 200;
   row updated.
   ```
4. The message “1 row updated.” indicates the update occurred correctly.
5. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   Commit complete.
   ```
6. The “Commit complete.” message indicates that the changes have been saved.

6. File Transfers to EDC DAAC

6.1 Introduction

This chapter instructs users on how to control and monitor EDC DAAC file transfers. It describes how to perform operations related to LPS L0R processing and how to

- View and edit LPS/EDC DAAC communication parameters stored in a string's database.
- Control automatic file availability notification.
- Determine the transfer status of a set of output files.
- Manage automatic output file deletion, including how to delete LPS output files manually.

6.2 LPS/EDC DAAC Communication Parameters Management

LPS/EDC DAAC communication parameters specify the host names, port numbers, user names, passwords, and other characteristics to be used in communications between the LPS and EDC DAAC.

NOTE

For Release 1, access to these parameters is possible only through SQL*Plus.

Each LPS string has its own set of LPS/EDC DAAC communication parameters. Parameter values must be set separately for each string.

6.2.1 Viewing LPS/EDC DAAC Communication Parameters

To view the current value of the LPS/EDC DAAC communication parameters:

1. Find the table and attribute names of the parameters you wish to view in Table 6–1.
2. Type the following to ORACLE SQL*Plus:

   ```
   SQL> SELECT Attributes FROM Tables;
   ```

   Attributes is a comma-separated list of the attribute names you wish to view. *Tables* is a comma-separated list of the tables containing the attributes.

Table 6–1. Table and Attribute Names for LPS/EDC DAAC Communication Parameters (1 of 2)

Parameter	Attribute Name	Range
LPS Port Number Port number on which the DDN server listens for connection requests from EDC DAAC	lps_configuration.lps_port_num	0..65535 NOTE: Port number should not be in use for any other Internet service on the LPS string.
EOS Core System (ECS) Hardware String ID Name of EDC DAAC host to which DANs should be sent in standard Internet name format, e.g., ecdaac2.cr.usgs.gov	lps_configuration.ecs_hardware_string_id	20-character maximum length
ECS Port Number Port number on which EDC DAAC host listens for connection requests from the LPS	lps_configuration.ecs_port_num	0..65535 NOTE: Port number should not be in use for any other Internet service on EDC DAAC string.
ECS User ID User name to be used for authentication when sending messages to the EDC DAAC host	lps_configuration.ecs_user_id	20-character maximum length
ECS Password Password to be used for authentication when sending messages to EDC DAAC host	lps_configuration.ecs_password	20-character maximum length
Authentication Retries Number of authentication failures to allow for EDC DAAC connection attempts before failing	valid_ldt_parms.num_auth_request	0..9999
DAA Timeout Specifies amount of time to wait for receipt of a DAA before timing out (seconds)	valid_ldt_parms.timeout_receive_daa	0..9999

Table 6–1. Table and Attribute Names for LPS/EDC DAAC Communication Parameters (2 of 2)

Parameter	Attribute Name	Range
DAN Timeout Specifies amount of time to wait while attempting to send a DAN before timing out (seconds)	valid_ldt_parms. timeout_send_dan_ attempt	0..9999
DDN Timeout Specifies amount of time to wait for receipt of a DDN before timing out (seconds)	valid_ldt_parms. timeout_receive_ddn	0..9999
DDA Retries Number of times to attempt to send a DDA	valid_ldt_parms.num_ send_dda_attempt	0..9999
DAA Timeout Specifies amount of time to wait while attempting to send a DAA before timing out (seconds)	valid_ldt_parms. timeout_send_dda	0..9999
Socket Read Interval Sleep time between socket reads (seconds)	valid_ldt_parms.read_ sleep_second	0..9999
DAN Retries Number of times to attempt to send a DAN	valid_ldt_parms.read_ num_send_dan_attempt	0..9999
Authentication Request Timeout Specifies amount of time to wait for acknowledgment of an authentication request before timing out (seconds)	valid_ldt_parms.read_ timeout_auth_request	0..9999

Example: To view the values of the ECS user ID, ECS password, and **TBD**:

1. For ECS user ID, the table name is “lps_configuration” and the attribute name is “ecs_user_id.” For ECS password, the table name is “lps_configuration” and the attribute name is “ecs_password.” For **TBD**, the table name is “valid_ldt_parms” and the attribute name is “read_sleep_second.”

2. Type the following to ORACLE SQL*Plus:

```
SQL> SELECT ecs_user_id, ecs_password, read_sleep_second
2>   FROM lps_configuration, valid_ldt_parms;
```

6.2.2 Setting LPS/EDC DAAC Communication Parameters

To modify an LPS/EDC DAAC connection parameter from ORACLE SQL*Plus:

1. Find the attribute and table name of the parameter you wish to modify in Table 6–1.
2. Type the following to SQL*Plus:

   ```
   SQL> UPDATE Table SET Attribute = Value;
   ```

   Table and *Attribute* are the attribute names found in step 1. *Value* is the new value you want to assign to the parameter.
3. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```
4. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   ```
5. Verify that the changes have been saved by noting that the following message appears:

   ```
   Commit complete.
   ```

Example: To set the ECS user ID to “LPS_String1”:

1. From Table 6–1, the ECS user ID table is “lps_configuration” and the attribute is “ecs_user_id.”
2. Type the following to SQL*Plus:

   ```
   SQL> UPDATE lps_configuration SET ecs_user_id = ‘LPS_String1’;
   1 row updated.
   ```
3. The message “1 row updated.” indicates the update has occurred successfully.
4. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   Commit complete.
   ```
5. The “Commit complete.” message indicates that the changes have been saved.

6.3 Automatic File Availability Notification Management

Normally, the LPS will automatically notify EDC DAAC that output files are available when L0R processing completes successfully. Users can, however, disable this feature and re-enable it later. While automatic file availability notification is disabled, a DAN is not sent to EDC DAAC whenever L0R processing produces a set of output files.

NOTE

For Release 1, controlling automatic file transfer is possible only through SQL*Plus.

The state of automatic notification is preserved even when the LPS system is shut down. When you bring the LPS software up again, the state (either enabled or disabled) will be the same as when it was last brought up.

When automatic notification is enabled after having been disabled, the LPS does not notify EDC DAAC of any file sets produced while automatic notification was disabled (**TBR**).

The state of automatic notification is independent on each LPS string. Enabling or disabling automatic notification on one string does not affect its state on other strings.

To disable automatic notification to EDC DAAC of output file availability, perform the following:

1. Type the following to SQL*Plus:

   ```
   SQL> UPDATE ldt_dan_transfer_state SET xfer_state = 0;
   ```

2. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```

3. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   ```

4. Verify that the changes have been saved by noting that the following message appears:

   ```
   Commit complete.
   ```

To enable automatic notification of EDC DAAC of output file availability, perform the following:

1. Type the following to SQL*Plus:

   ```
   SQL> UPDATE ldt_dan_transfer_state SET xfer_state = 1;
   ```

2. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```

3. Save the change by typing the following to SQL*Plus:

```
SQL> COMMIT;
```

4. Verify that the changes have been saved by noting that the following message appears:

```
Commit complete.
```

To determine the status of automatic file availability notification, type the following to SQL*Plus:

```
SQL> SELECT xfer_state FROM ldt_dan_transfer_state;
```

If the reported value of xfer_state is 0, file availability notification is disabled. If the reported value is 1, file availability notification is enabled.

To monitor automatic notification, view the status and error messages generated by the senddan process, which notifies EDC DAAC that output files are available for transfer. To view status and error messages generated by senddan, set up a real-time display of the LPS Journal file (see Section 2.7.2).

6.4 DDN Server Management

The DDN server, rcvddn, accepts and processes DDNs from EDC DAAC. The DDN server is a daemon process that normally executes in the background. Users can control automatic DDN receipt and processing by terminating or restarting the DDN server.

NOTE

For Release 1, the DDN server can be controlled only through IRIX shell commands.

There is a separate DDN server on each LPS string. Changing the status of the DDN server on one string does not affect the status of the DDN servers on the other strings.

6.4.1 Checking Whether the DDN Server Is Active

To determine whether the DDN server is active, hold the mouse button down to pull down the **FileMgmt** menu. If the **Stop DDN Server...** command appears, the DDN server is active. If the **Start DDN Server...** command appears, the DDN server is not active.

6.4.2 Stopping the DDN Server

WARNING

For Release 1, stopping the DDN server while it is processing a DDN will cause it to terminate prematurely, possibly leaving the LPS in an inconsistent state.

Before stopping the DDN server, verify that the DDN server is not currently processing a DDN by **TBD**.

To stop the DDN server, select the **FileMgmt>Stop DDN Server...** command and click the OK button in the confirmation dialog.

6.4.3 Starting the DDN Server

To start the DDN server, select the **FileMgmt>Start DDN Server...** command and click the OK button in the confirmation dialog.

6.5 Checking the Status of a Set of Output Files

To determine the status of a set of output files, first determine whether the LPS has notified EDC DAAC that the files are available (by sending a DAN) and whether EDC DAAC has acknowledged receipt of the notification (by returning a DAA). If EDC DAAC has been notified and has acknowledged, determine whether or not the files have been transferred successfully.

6.5.1 Checking Whether EDC DAAC Has Been Notified

To determine whether EDC DAAC has been notified that a set of output files is ready for transfer:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how, read Appendix E, Sections E.1.1 and E.1.2.
2. Determine whether a DAN was sent for the set of output files by typing the following to SQL*Plus:

   ```
   SQL> SELECT dan_status FROM ldt_dan_info
   2>   WHERE contact_sequence_id = Contact Identifier
   3>   AND file_version_number = File Version Number ;
   ```

   Contact Identifier is the contact ID for the contact. *File Version Number* is the file version number.
3. Use Table 6–2 to look up the meaning of the value reported by SQL*Plus.

Example: Determine the status of the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Section E.1.2, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.

Table 6–2. Meaning of dan_status Attribute Values

Value	Meaning
0	EDC DAAC is being notified
1	EDC DAAC has not been notified because automatic notification was disabled when this set of output files was ready for transfer.
2	EDC DAAC has been notified and the LPS is waiting for acknowledgment from EDC DAAC.
3	EDC DAAC has been notified and has also acknowledged the notification.
4	EDC DAAC has been notified, has not acknowledged, and the maximum waiting time for an acknowledgment has been exceeded.

2. Determine whether a DAN was sent for the set of output files by typing the following to SQL*Plus:

```
SQL> SELECT dan_status FROM ldt_dan_info
2>   WHERE contact_sequence_id = 86214
3>   AND file_version_number = 0;
DAN_STATUS
----------
         3
```

   Table 6–2 indicates that a value of 3 means EDC DAAC has been notified and has acknowledged the notification.

6.5.2 Checking Whether EDC DAAC Has Transferred the Files

To determine whether EDC DAAC has transferred a set of files:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, read Sections E.1.1 and E.1.2.
2. Verify that EDC DAAC has been notified of its availability as described in Section 6.4.1. If EDC DAAC has not been notified that the files are available, EDC DAAC has not yet transferred the files.
3.

Fetch the status of the output files from the LPS database by typing the following to SQL*Plus:

```
SQL> SELECT file_set_transfer_status FROM ldt_file_set_info
2>   WHERE contact_sequence_id = Contact Identifier
3>   AND file_version_number = File Version Number;
```

Contact Identifier and *File Version Number* are the contact ID and file version number, respectively, determined in step 1.

4. Use Table 6–3 to look up the meaning of the value reported by SQL*Plus.

Table 6–3. Meaning of file_set_transfer_status Attribute Values (TBR)

Value	Meaning
0	The files have been successfully transferred.
1	EDC DAAC failed in its last attempt to transfer any files. No files have been transferred.
2	EDC DAAC has not yet sent a DDN. No files have been transferred.
3	EDC DAAC has successfully transferred some of the files. Some of the files have not been transferred.

5. If the value of file_set_transfer_status is 3 (some of the files have been successfully transferred and others have not), you can determine the disposition of each file by typing the following to SQL*Plus:

```
SQL> COLUMN file_path HEADING 'Path' FORMAT A25
SQL> COLUMN file_name HEADING 'Name' FORMAT A22
SQL> COLUMN file_xfer_disposition HEADING 'Disp' FORMAT 000
SQL> SELECT file_path, file_name, file_xfer_disposition
3>   FROM ldt_file_set_info f, sub_intv s, lps_file_info i
4>   WHERE f.contact_sequence_id = 86214
5>   AND f.file_version_number = 0
6>   AND f.contact_sequence_id = s.contact_sequence_id
7>   AND f.file_version_number = s. file_version_number
```

```
8>   AND s.sub_intv_sequence_id = i.sub_intv_sequence_id
9>   ORDER BY file_path, file_name;
```

6. Use Table 6–4 to look up the meaning of the file_transfer_disposition reported by SQL*Plus for each file.

Example: Determine whether the file set in the previous example has been transferred to EDC DAAC.

1. Using one of the methods described in Section E.1.2, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.

2. As determined in the previous example, EDC DAAC has been notified and has acknowledged.

3. Type the following to SQL*Plus:

```
SQL> SELECT file_set_transfer_status FROM ldt_file_set_info
2>   WHERE contact_sequence_id = 86214
3>   AND file_version_number = 0;
file_set_transfer_status
------------------------
                       3
```

4. In Table 6–3, a file_set_transfer_status value of 3 indicates that some of the files have been transferred and others have not.

5. Type the following to SQL*Plus:

```
SQL> COLUMN file_path HEADING ‘Path’ FORMAT A25
SQL> COLUMN file_name HEADING ‘Name’ FORMAT A22
SQL> COLUMN file_xfer_disposition HEADING ‘Disp’ FORMAT 000
SQL> SELECT file_path, file_name, file_xfer_disposition
3>   FROM ldt_file_set_info f, sub_intv s, lps_file_info i
4>   WHERE f.contact_sequence_id = 86214
5>   AND f.file_version_number = 0
6>   AND f.contact_sequence_id = s.contact_sequence_id
7>   AND f.file_version_number = s. file_version_number
8>   AND s.sub_intv_sequence_id = i.sub_intv_sequence_id
9>   ORDER BY file_path, file_name;
```

Table 6–4. Meaning of file_xfer_disposition Attribute Values (TBR)

Value	Meaning
0	File has been successfully transferred
1	Network failure
2	Unable to establish FTP connection
3	Host denied access
4	All file groups/files not found
5	FTP failure – too many errors in file transfer
6	Post-transfer doublecheck failed
7	FTP command error
242	Data provider threshold exceeded
243	System threshold exceeded
244	Resource allocation failure
245	Recovery failure
246	Database access error
247	Insufficient number of metadata files
248	Insufficient number of science files
249	Insufficient number of files
250	Data conversion failure
251	Metadata checking failure
252	Unknown data type
253	Undeterminable data type
254	File input/output error
255	Data archive error

```
Path                        Name                      Disp
--------------------------- ------------------------- ----
/u01/st/86214/0/3624        L71EDC219821109000.B60     0
/u01/st/86214/0/3624        L71EDC219821109000.B70     0
/u01/st/86214/0/3624        L71EDC219821109000.B81     0
/u01/st/86214/0/3624        L71EDC219821109000.B82     0
/u01/st/86214/0/3624        L71EDC219821109000.CAL   254
/u01/st/86214/0/3624        L71EDC219821109000.MSD     0
/u01/st/86214/0/3624        L71EDC219821109000.MTA     0
/u01/st/86214/0/3624        L71EDC219821109000.PCD     0
/u01/st/86214/0/3625        L71EDC219821109010.B60     0
/u01/st/86214/0/3625        L71EDC219821109010.B70     0
/u01/st/86214/0/3625        L71EDC219821109010.B81     5
/u01/st/86214/0/3625        L71EDC219821109010.B82     0
/u01/st/86214/0/3625        L71EDC219821109010.CAL     0
/u01/st/86214/0/3625        L71EDC219821109010.MSD     0
/u01/st/86214/0/3625        L71EDC219821109010.MTA     0
/u01/st/86214/0/3625        L71EDC219821109010.PCD     0
```

6. Table 6–4 indicates that all files have transferred except for L71EDC219821109000.CAL, which failed because of file input/output errors at EDC DAAC, and L71EDC219821109010.B81, which failed because of FTP errors.

6.6 Checking Whether EDC DAAC Is Transferring Files

To determine whether EDC DAAC is currently transferring files from a string:

1. Find the EDC DAAC host name in the LPS database by typing the following to SQL*Plus:

   ```
   SQL> SELECT edc_host_name FROM lps_configuration;
   ```

2. Determine whether there is an active FTP connection by typing the following to an IRIX shell:

   ```
   % netstat | grep EDC DAAC Host Name | grep ftp
   ```

3. If the FTP connection to the EDC DAAC host is active, this command will produce a line describing the state of the connection:

   ```
   tcp 0 0 lps001.cr.edc.gov.ftp edcdaac1.cr.edc.gov.1098 ESTABLISHED
   ```

4. For a description of the line's meaning, consult the netstat(1) section of *IRIX 5.3 Reference Manual (*Reference 2).

6.7 Checking the Contents of LPS/EDC DAAC Messages

Normally, LPS/EDC DAAC communication messages are processed and discarded. However, the LPS software is able to save all LPS/EDC DAAC communications messages for review.

NOTE

Saving LPS/EDC DAAC communication messages is controlled independently on each LPS string. You must set the LDT_SAVE_MESSAGE environment variable on each string on which you want to save messages.

To activate automatic saving of LPS/EDC DAAC communications messages, use the "setenv" IRIX shell command to set the value of LDT_SAVE_MESSAGE to 1. For example:

```
% setenv LDT_SAVE_MESSAGE 1
```

You must issue this command BEFORE starting up the LPS software.

To deactivate this feature, exit the LPS GUI and use the "unsetenv" IRIX shell command to remove LDT_SAVE_MESSAGE from the environment:

```
% unsetenv LDT_SAVE_MESSAGE
```

The change will be effective when you restart the LPS software.

NOTE

Whether saving LPS/EDC DAAC communication messages is enabled or disabled is determined when the LPS software is started. Subsequent changes will not be reflected in LPS software behavior until the LPS software is restarted.

When automatic LPS/EDC DAAC communications saving is enabled, the LPS software will write DANs and DAAs to the directory indicated by the LPS_DANFILE_PATH environment variable (normally "$LPS_HOME/DAN"). It will write DDNs, DDAs, and authentication messages to the directory indicated by the LPS_DDNFILE_PATH environment variable (normally "$LPS_HOME/DDN"). The file names have the format

TTT_NNNNNNNNNN_MM.msg

where TTT is the message type (DAN, DAA, DDN, or DDA), NNNNNNNNNN is the DAN ID associated with the communication, and MM is an incrementing message number (**TBR**). To determine the DAN ID associated with a particular instance of L0R output files:

1. Find the contact ID and the file version number of the L0R processing instance of interest. To learn how to do so, read Appendix E, Sections E.1.1 and E.1.2.
2.

Type the following to SQL*Plus:

```
SQL> SELECT dan_sequence_num FROM ldt_dan_info
2>   WHERE contact_sequence_id = Contact Identifier
3>   AND file_version_number = File Version Number;
```

LPS/EDC DAAC messages are saved in American Standard Code for Information Interchange (ASCII) files. The files can be viewed using any of the IRIX file display utilities, such as more(1) or page(1) and may be modified using any text editor, such as vi. See Reference 2 for details on these utilities.

6.8 Managing Automatic Output File Deletion

This section describes how to control the LPS software's automatic file deletion capabilities. It describes how to mark file sets for retention so that they will not be deleted automatically. It also describes how to delete LPS output files manually.

NOTE

For Release 1, managing automatic output file deletion is possible only through SQL*Plus.

The LPS software may delete a subset of the files from an instance of L0R processing, but only entire file sets can be marked for retention.

6.8.1 Marking a File Set for Retention

To mark a set of output files for retention:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, see Appendix E, Sections E.1.1 and E.1.2.
2. Mark the file set for retention by typing the following to SQL*Plus:

   ```
   SQL> UPDATE processing_version_info SET retention_status = 1
   2>   WHERE contact_sequence_id = Contact Identifier
   3>   AND file_version_number = File Version Number;
   ```

3. Verify that the update occurred correctly by noting that the following message appears:

   ```
   1 row updated.
   ```

4. Save the change by typing the following to SQL*Plus:

   ```
   SQL> COMMIT;
   ```

5. Verify that the changes have been saved by noting that the following message appears:

```
Commit complete.
```

Example: Mark for retention the file set produced by the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Section E.1.2, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.
2. Mark the file set for retention by typing the following to SQL*Plus:

```
SQL> UPDATE processing_version_info SET retention_status = 1
2>   WHERE contact_sequence_id = 86214
3>   AND file_version_number = 0;
1 row updated.
```

3. The message "1 row updated." indicates that the update was successful.
4. Save the change by typing the following to SQL*Plus:

```
SQL> COMMIT;
Commit complete.
```

5. The "Commit complete." message indicates that the changes have been saved.

6.8.2 Determining Whether a File Set Has Been Marked for Retention

To determine whether a file set has been marked for retention:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, read Appendix E, Sections E.1.1 and E.1.2.
2. Type the following to SQL*Plus:

```
SQL> SELECT retention_status FROM processing_version_info
2>   WHERE contact_sequence_id = Contact Identifier
3>   AND file_version_number = File Version Number;
```

3. If the value reported is 1, the file has been marked for retention. If the value reported is 0, the file has not been marked for retention.

Example: Determine whether the file set in the previous example has been marked for retention.

1. As noted in the previous example, the contact ID is 86214 and the file version number is 0.
2.

Type the following to SQL*Plus:

```
SQL> SELECT retention_status FROM processing_version_info
2>   WHERE contact_sequence_id = 86214
3>   AND file_version_number = 0;
PROCESSING_VERSION_INFO
-----------------------
                      0
```

3. The value reported is 0, so the file set has not been marked for retention.

6.9 Deleting L0R Output Files

For Release 1, the LPS software does not support output file deletion. You must use other system software to delete output files. The LPS software maintains a record of the deletion status (deleted or not deleted) for every output file in the LPS database. When you delete output files, you must also make the appropriate updates to the LPS database.

WARNING

Deleting L0R output files without making the appropriate updates in the LPS database will render the LPS database inconsistent and may cause run-time errors in LPS software.

You can delete an output file set, an output file group, or individual output files.

6.9.1 Deleting a File Set

To delete a set of output files:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, read Appendix E, Sections E.1.1 and E.1.2.
2. Delete all files in the set by typing the following to an IRIX shell:

   ```
   % cd $LPS_OUTFILE_PATH/Contact Identifier/File Version Number
   % rm -r *
   ```

3. Mark each file group in the set as deleted by typing the following to SQL*Plus:

   ```
   SQL> UPDATE ldt_file_group_info
   2>   SET file_group_deletion_status = 1
   3>   WHERE sub_intv_sequence_id IN
   4>   (SELECT sub_intv_sequence_id FROM sub_intv
   ```

```
5>     WHERE contact_sequence_id = Contact Identifier
6>     AND file_version_number = File Version Number);
```

4. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

5. Mark each file in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id IN
4>   (SELECT sub_intv_sequence_id FROM sub_intv
5>     WHERE contact_sequence_id = Contact Identifier
6>     AND file_version_number = File Version Number);
```

6. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

7. Commit the updates by typing the following to SQL*Plus:

```
SQL> COMMIT;
```

8. Verify that the commit completed successfully by noting that the following message appears:

```
Commit complete.
```

Example: Delete the file set produced by the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Section E.1.2, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.

2. Delete all files in the set by typing the following to an IRIX shell:

```
% cd $LPS_OUTFILE_PATH/86214/0
% rm -r *
```

3. Mark each file group in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_group_info
2>   SET file_group_deletion_status = 1
3>   WHERE sub_intv_sequence_id IN
4>   (SELECT sub_intv_sequence_id FROM sub_intv
```

```
5>     WHERE contact_sequence_id = 86214
6>     AND file_version_number = 0);
1 row updated.
```

4. The message “1 row updated.” indicates that the update occurred correctly.

5. Mark each file in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id IN
4>   (SELECT sub_intv_sequence_id FROM sub_intv
5>     WHERE contact_sequence_id = 86214
6>     AND file_version_number = 0);
1 row updated.
```

6. The message “1 row updated.” indicates that the update occurred correctly.

7. Commit the updates by typing the following to SQL*Plus:

```
SQL> COMMIT;
Commit complete.
```

8. The message “Commit complete.” indicates that the update has been saved to the database.

6.9.2 Deleting a File Group

WARNING

Deleting the last file group of a file set without updating the file set information in the database can leave the LPS database in an inconsistent state. If you are deleting the last file group of a file set, follow the procedure for deleting a file set rather than the procedure defined here.

To delete a file group:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, read Appendix E, Sections E.1.1 and E.1.2.

2. Find the LPS database subinterval ID for the file group of interest. To learn how to retrieve the subinterval ID from the database, read Section E.1.3.

3.

Delete all files in the set by typing the following to an IRIX shell:

```
% cd $LPS_OUTFILE_PATH/Contact Identifier/File Version Number \
? /Subinterval Identifier
% rm -r *
```

4. Mark each file group in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_group_info
2>   SET file_group_deletion_status = 1
3>   WHERE sub_intv_sequence_id = Subinterval Identifier;
```

5. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

6. Mark each file in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id = Subinterval Identifier;
```

7. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

8. Commit the updates by typing the following to SQL*Plus:

```
SQL> COMMIT;
```

9. Verify that the commit completed successfully by noting that the following message appears:

```
Commit complete.
```

Example: Delete the file group for the first subinterval in the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Section E.1.1, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.

2. Using one of the methods described in Section E.1.3, the subinterval ID is determined to be 341576.

3. Delete all files in the set by typing the following to an IRIX shell:

```
% cd $LPS_OUTFILE_PATH/86214/0/341576
% rm -r *
```

4. Mark each file group in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_group_info
2>   SET file_group_deletion_status = 1
3>   WHERE sub_intv_sequence_id = 341576;
1 row updated.
```

5. The message “1 row updated.” indicates the update occurred correctly.
6. Mark each file in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id = 341576;
1 row updated.
```

7. The message “1 row updated.” indicates the update occurred correctly.
8. Commit the updates by typing the following to SQL*Plus:

```
SQL> COMMIT;
Commit complete.
```

9. The message “Commit complete.” indicates the commit occurred correctly.

6.9.3 Deleting Individual Files

WARNING

Deleting the last file of a file group without updating the file group information in the database can leave the LPS database in an inconsistent state. If you are deleting the last file of a file group, follow the procedure for deleting a file group rather than the procedure defined here.

To delete an LPS output file:

1. Find the LPS database contact ID and the file version number for the processing instance of interest. To learn how to do so, read Appendix E, Sections E.1.1 and E.1.2.
2. Find the LPS database subinterval ID for the file group of interest. To learn how to retrieve the subinterval ID from the database, read Section E.1.3.
3.

Delete the file by typing the following to an IRIX shell:

```
% rm $LPS_OUTFILE_PATH/Contact Identifier/File Version Number \
? /Subinterval Identifier/File Name
```

4. Mark the file as deleted in the LPS database by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id = Subinterval Identifier
4>   AND file_name = 'File Name';
```

5. Verify that the update occurred correctly by noting that the following message appears:

```
1 row updated.
```

6. Commit the update by typing the following to SQL*Plus:

```
SQL> COMMIT;
```

7. Verify that the commit completed successfully by noting that the following message appears:

```
Commit complete.
```

Example: Delete the file L71EDC219821109010.CAL for the first subinterval in the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Section E.1.1, the contact ID is determined to be 86214. The file version number is 0 because this is the first instance of processing the contact.

2. Using one of the methods described in Section E.1.3, the subinterval ID is determined to be 341576.

3. Delete the file in the set by typing the following to an IRIX shell:

```
% rm $LPS_OUTFILE_PATH/86214/0/341576/L71EDC219821109010.CAL
```

4. Mark the file group in the set as deleted by typing the following to SQL*Plus:

```
SQL> UPDATE ldt_file_info
2>   SET file_deletion_status = 1
3>   WHERE sub_intv_sequence_id = 86214
4>   AND file_name = 'L71EDC219821109010.CAL';
1 row updated.
```

5. The message "1 row updated." indicates the update occurred correctly.

6. Commit the updates by typing the following to SQL*Plus:

```
SQL> COMMIT;
Commit complete.
```

7. The message “Commit complete.” indicates the commit occurred correctly.

7. Report Generation

NOT IMPLEMENTED IN RELEASE 1

8. Testing the LPS

NOT IMPLEMENTED IN RELEASE 1

9. LPS Maintenance

9.1 Introduction

This chapter explains how to perform routine LPS maintenance activities. These activities include backing up the LPS database; purging the LPS database of obsolete records; purging the LPS Journal file; deleting unneeded trouble files, saved LPS/EDC DAAC messages, and saved reports; and deleting temporary files left after an abnormal LPS process termination. Maintenance functions should not be performed while any LPS processing is in progress.

This chapter does not explain how to perform hardware and system software maintenance or hardware or software upgrades. It also does not include detailed instructions for recovering from catastrophic LPS failures, such as system crashes. For instructions on these subjects, consult *LPS Operations and Maintenance Manual* and *LPS Installation Procedure* (References 3 and 4, respectively).

9.2 Backing Up the LPS Database

NOT IMPLEMENTED IN RELEASE 1

9.3 Purging the LPS Database

The LPS database contains records describing each contact, each L0R processing run, and each output file's transfer to EDC DAAC. The LPS software does not automatically delete these records. This allows the user to save Q&A information indefinitely, but also requires periodic removal of unneeded records. The SQL script in Figure 9–1 can be used to remove records associated with a contact captured before a date that the user specifies.

To use the script to purge the LPS database:

1. Use any text editor to create a file on the LPS string you wish to purge containing the statements in Figure 9–1. Let *filename* denote the name of the file you have created (an .sql extension is required).
2. Start up SQL*Plus from the directory containing the file by typing:

   ```
   % sqlplus /
   ```

3. Type the following to SQL*Plus:

   ```
   SQL> start filename (without .sql extension)
   Enter date as MM/DD/YY: MM/DD/YY
   SQL*Plus processing messages
   ```

Executing the script requires select and delete permission on the LPS database tables, as well as create table permission for the LPS database instance.

```
/* Delete Q&A records for contacts earlier than specified date */

/* Accept the date */

accept the Date prompt 'Enter date as MM/DD/YY: '

/* Create a tmp table to hold the IDs of items to be deleted */

create table dbpurge_tmp ( cid number(10), sid number(10) );
insert into dbpurge_tmp
  select sub_intv.contact_sequence_id, sub_intv_sequence_id
     from sub_intv, rdc_acct
     where sub_intv.contact_sequence_id = rdc_acct.contact_sequence_id
       and rdc_acct.actual_stop_time < to_date('&theDate','MM/DD/YY');

/* Contact Accounting */

delete from rdc_acct where contact_sequence_id in (select cid from dbpurge_tmp);

/* L0R Q&A */

delete from processing_version_info
  where contact_sequence_id in (select cid from dbpurge_tmp);
delete from rdp_acct
  where contact_sequence_id in (select cid from dbpurge_tmp);
delete from sub_intv
  where contact_sequence_id in (select cid from dbpurge_tmp);
delete from mfp_acct
  where sub_intv_sequence_id in (select sid from dbpurge_tmp);

/* File Transfer Accounting */

delete from ldt_dan_info
  where sub_intv_sequence_id in (select sid from dbpurge_tmp);
delete from ldt_file_group_info
  where sub_intv_sequence_id in (select sid from dbpurge_tmp);
delete from lps_file_info
  where sub_intv_sequence_id in (select sid from dbpurge_tmp);
delete from ldt_file_set_info
  where contact_sequence_id in (select cid from dbpurge_tmp);

/* Old CCSDS Parameter Sets no longer needed for reporting */
/* Note that this assumes rdp_acct has already been purged */

delete from valid_ccsds_parms
  where ccsds_parms_id not in (select unique ccsds_parms_id from rdp_acct)
        and insertion_time != (select max(insertion_time) from valid_ccsds_parms);

/* Drop the tmp table */

drop table dbpurge_tmp;
```

Figure 9–1. LPS Database Purge Script

9.4 Purging the LPS Journal

The LPS Journal file grows as messages are regularly added. The file can consume excessive amounts of disk space. Displaying the LPS Journal for browsing will require unacceptably long times to format the file for display. Therefore, periodically purge the LPS Journal file by deleting all of its contents and starting with a new empty file.

WARNING

This operation deletes the contents of the LPS Journal file. If you wish to save the LPS Journal file contents for future reference, make a copy of the file before purging it.

To purge the LPS Journal file, type the following command to an IRIX shell:

```
% touch `grep local0.debug /etc/syslog.conf | awk '{print $NF}'`
```

This command requires write permission on the LPS Journal file.

NOTE

Each LPS string has its own Journal file. To purge all Journal files, you must issue the touch command on each LPS string.

9.5 Purging Trouble Files

Trouble files produced during L0R processing are not automatically deleted. They will consume excessively large amounts of disk space if allowed to accumulate. Therefore, periodically purge the directories containing trouble files.

WARNING

This operation deletes trouble files. If you wish to save them for future reference, copy them to tape before purging the directories.

To purge the directories containing trouble files, type the following command to an IRIX shell:

```
% rm $LPS_TROUBLEFILE_PATH/*TroubleFile*
```

This command requires write permission on all trouble files.

NOTE

Each LPS string has its own directory for storing trouble files. To purge all trouble files, you must issue the `% rm $LPS_TROUBLEFILE_PATH/*TroubleFile*` command on each LPS string.

9.6 Purging Saved LPS/EDC DAAC Communications Messages

Files containing saved messages between the LPS and EDC DAAC are not automatically deleted. If saving LPS/EDC DAAC communications is enabled, these files may consume

excessively large amounts of disk space if allowed to accumulate. Therefore, periodically purge the directories containing saved communications messages.

WARNING

This operation deletes the files. If you wish to save them for future reference, copy them to tape before purging the directories.

To purge the directories containing trouble files, type the following commands to an IRIX shell:

```
% rm $LPS_DANFILE_PATH/DAN_*MM.msg
% rm $LPS_DANFILE_PATH/DAA_*MM.msg
% rm $LPS_DDNFILE_PATH/DDN_*MM.msg
% rm $LPS_DDNFILE_PATH/DDA_*MM.msg
```

This command requires write permission on all LPS/EDC DAAC saved messages files.

NOTE

Each LPS string has its own directory for storing LPS/EDC DAAC messages. To purge all LPS/EDC DAAC message files, you must issue the commands above on each LPS string.

9.7 Purging the Reports Directory

NOT IMPLEMENTED IN RELEASE 1

9.8 Clearing the tmp Directory

Some LPS processes create temporary files that are normally deleted when the process exits. If the process terminates abnormally (for example, as the result of a system crash), the file may still exist. Such residual files may make it impossible to perform certain functions. Therefore, delete residual temporary files after any abnormal LPS software termination.

To delete residual temporary files, type the following to an IRIX shell:

```
% rm $LPS_TEMPFILE_PATH/*
% rm /tmp/rdc_PIDTempFile
```

This command requires write permission on the temporary files.

NOTE

Each LPS string has its own directory for storing temporary files. To purge residual, temporary files on multiple strings, you must issue the command above on each LPS string.

9.9 LPS File Consistency

NOT IMPLEMENTED IN RELEASE 1

10. LPS GUI Detailed Reference

10.1 Introduction

This chapter explains each LPS GUI command. Explanations are arranged according to the main menu's top-level pulldown menus. Figure 10–1 shows a view of the LPS main menu bar. Each explanation includes a description of what the menu option does, what additional information you may need to provide, what default values are used when you do not supply additional information, and explanations of any additional dialogs displayed by the LPS GUI.

This chapter does not provide step-by-step explanations for performing LPS procedures. To find out how to carry out a specific procedure, refer to Sections 3 through 9.

Figure 10–1. LPS GUI Main Menu

10.2 Setup Menu

The Setup menu (Figure 10–2) contains commands to view and edit information on LPS string configuration, L0R processing parameters, and L0R processing error reporting thresholds.

10.2.1 Set Capture Source...

NOT IMPLEMENTED IN RELEASE 1

Figure 10–2. LPS GUI Setup Menu

10.2.2 Ingest Contact Schedule...

NOT IMPLEMENTED IN RELEASE 1

10.2.3 View/Edit Contact Schedule...

NOT IMPLEMENTED IN RELEASE 1

10.2.4 Ingest IAS Parameter File...

NOT IMPLEMENTED IN RELEASE 1

10.2.5 View/Edit IAS Parameters...

NOT IMPLEMENTED IN RELEASE 1

10.2.6 View/Edit Other L0R Parameters...

NOT IMPLEMENTED IN RELEASE 1

10.2.7 View/Edit L0R Thresholds...

NOT IMPLEMENTED IN RELEASE 1

10.2.8 View/Edit Output File Transfer Config...

NOT IMPLEMENTED IN RELEASE 1

10.3 Control Menu

The Control menu (Figure 10–3) contains commands to perform LPS functions such as data capture, copying to tape, and L0R processing, as well as commands to enable or disable LPS automatic processes.

Figure 10–3. LPS GUI Control Menu

10.3.1 Start Capture...

Selecting this option causes the LPS software to begin capturing raw wideband data. The LPS GUI displays the Start Capture dialog (Figure 10–4), requesting further information about the contact or test data transmission to be captured as described in Table 10–1. Clicking CANCEL cancels the operation. Clicking OK starts data capture immediately using the information the user has provided in the dialog.

The Start Capture command changes to **Stop Capture…** whenever the LPS is currently capturing data. When you select this option, the LPS GUI displays the Stop Capture Confirmation dialog (Figure 10–5). Clicking YES causes the current capturing to be halted. Clicking NO cancels the operation; the LPS software continues to capture data.

Figure 10–4. LPS GUI Start Capture Dialog

Table 10–1. Additional Information for the Start Capture Command

Item	Description
Capture Source	The source of the data to be captured. The default is the current capture source in the LPS string's configuration. You can select an alternate source by holding down the mouse button over the field and selecting one of the options. The options and their meanings are as follows: · 1I = X-band 1 I channel · 1Q = X-band 1 Q channel · 2I = X-band 2 I channel · 2Q = X-band 2 Q channel · 3I = X-band 3 I channel · 3Q = X-band 3 Q channel · TB = LGS bit error rate tester · T1 = Test data from LPS string 1 · T2 = Test data from LPS string 2 · T3 = Test data from LPS string 3 · T4 = Test data from LPS string 4 · T5 = Test data from LPS backup/development string
Isolate Capture Process	NOT IMPLEMENTED IN RELEASE 1
Suspend L0R Processing	NOT IMPLEMENTED IN RELEASE 1
Stop Time	The time to stop capturing data. The default is 14 minutes—the maximum possible duration. You may enter another earlier time by typing the value (format = YYYY:DDD:HH:MM:SS) into the field.

10.3.2 Start Level 0R Processing...

Selecting this option causes the LPS software to begin processing a file containing raw wideband data to L0R. The LPS GUI displays the Start Level 0R Processing dialog (Figure 10–6), requesting further information about the contact or test data transmission to be processed as described in Table 10–2. Clicking CANCEL cancels the operation. Clicking OK starts L0R processing immediately using the information the user has provided in the dialog.

10.3.3 Stop Level 0R Processing...

Selecting this option causes the LPS software to stop a specified instance of L0R processing. The LPS GUI displays the Stop Level 0R Processing dialog (Figure 10–7), displaying a list of the raw data files being processed along with the file version number assigned to this instance of its processing. Select the instance of L0R processing to terminate. The selected line is highlighted. By default, the first line is selected. Clicking CANCEL cancels the operation. Clicking OK causes the LPS GUI to display the confirmation dialog (Figure 10–8). Clicking OK a second time terminates the selected instance of L0R processing. Clicking CANCEL cancels the operation and returns the user to the Stop Level 0R Processing dialog.

Figure 10–5. LPS GUI Stop Capture Confirmation Dialog

This option is enabled only when the LPS software is currently processing at least one raw wideband data file to L0R.

10.3.4 DAN Transfer State...

NOT IMPLEMENTED IN RELEASE 1

10.3.5 Start Copy to Tape..

NOT IMPLEMENTED IN RELEASE 1

Figure 10–6. LPS GUI Start Level 0R Processing Dialog

10.3.6 Start Restage...

NOT IMPLEMENTED IN RELEASE 1

10.3.7 Enable Auto Capture...

NOT IMPLEMENTED IN RELEASE 1

Table 10–2. Additional Information for the Start Level 0R Processing Command

Item	Description
Raw Data File	The file containing raw wideband data to be processed. The dialog displays identifying information for each raw wideband data file stored on the string's raw data storage array. Select the file by clicking on any of the fields describing it. The selected file is highlighted. When the dialog is first displayed, the first file in the list is selected by default.
File Version Number	The right-most field for each raw data file contains the file version number that will be assigned to this instance of L0R processing for this file. By default, the value is one greater than the highest value recorded for this raw wideband data file in the string's database. Enter another value if, for instance, this raw data has already been processed with this file version number on another string. NOTE: If the file version number supplied is greater than the default, the LPS GUI will issue a warning dialog asking the user to confirm the selection. Once confirmed, the LPS software will use the specified file version number. The LPS software will not accept a file version number that has already been used for another instance of L0R processing on this file on this string.
Delete the raw data file after processing	When selected, this option will cause the raw data file to be deleted automatically from the string's raw data storage array after it has been successfully processed. The default is NOT to delete the file. NOTE: The file will be deleted only if both L0R processing completes successfully and the raw data file has been copied to the 30-day store. Otherwise, the file will remain online.
Delete the output files after transmission	When selected, this option will cause the output files to be deleted automatically from the string's output file storage array after they are successfully transferred to EDC DAAC. The default is to delete the files automatically.

10.3.8 Enable Auto Tape Saving...

NOT IMPLEMENTED IN RELEASE 1

10.3.9 Enable Auto Level 0R

NOT IMPLEMENTED IN RELEASE 1

Figure 10–7. LPS GUI Stop Level 0R Processing Dialog

Figure 10–8. LPS GUI Stop Level 0R Processing Confirmation Dialog

10.4 Reports

The Reports menu (Figure 10–9) contains commands to generate LPS Q&A reports.

Figure 10–9. LPS GUI Reports Menu

10.4.1 Data Receive Summary...

NOT IMPLEMENTED IN RELEASE 1

10.4.2 LPS Quality and Accounting...

NOT IMPLEMENTED IN RELEASE 1

10.4.3 File Transfer Summary...

NOT IMPLEMENTED IN RELEASE 1

10.5 Test Menu

The Test menu (Figure 10–10) contains a single command to play back a captured raw wideband data file.

Figure 10–10. LPS GUI Test Menu

10.5.1 Send Test Data...

NOT IMPLEMENTED IN RELEASE 1

10.6 Monitor Menu

The Monitor menu (Figure 10–11) contains commands to view LPS software status and error messages, either by displaying messages as they are generated or by displaying the messages stored in the LPS Journal file.

Figure 10–11. LPS GUI Monitor Menu

10.6.1 Display LPS Journal File...

Selecting this option causes the LPS software to create a window (Figure 10–12) displaying the formatted contents of the LPS Journal file. This window contains menus allowing the user to customize the display format; filter the messages displayed by priority or other characteristics; sort the messages by priority or time; print, mail, or save selected messages; and perform other functions. Online help for the window's options is available through the window's **Help** menu. Additional information is available from sysmon(1) in Reference 2.

NOTE

The LPS Journal file display window maps the eight LPS message priorities to four categories of priority. See Section 2.7.2 for details.

Figure 10–12. LPS GUI Journal File Display Window

To terminate the display, select **File>Quit** from the window's menu bar.

10.6.2 Display Operations Messages...

Selecting this option causes the LPS software to create a window in which LPS status and error messages will be displayed as they are generated. The LPS GUI displays the Display Operational Messages dialog (Figure 10–13), requesting the user to select a set of message priorities for display. The message display window will display only the priorities selected. By default, all message priorities except Debug are selected for display (see Table 2–1 for an explanation of LPS message priorities and their meanings). Clicking CANCEL cancels the operation; no display is created. Clicking OK creates a display window (Figure 10–14) for messages of the selected priorities. The window can be moved, resized, or iconified and the display font can be changed. For details, see Chapter 8, "Managing Windows," in *IRIX Essentials* (Reference 5) and the xwsh(1G) in Reference 2.

To terminate the display, move the mouse to the window's menu bar, hold down the right mouse button, and select **Quit** from the pulldown menu.

10.7 File Management Menu

The File Management menu (Figure 10–15) contains commands related to LPS output file handling. Commands in this menu allow the user to send a DAN to EDC DAAC, indicate that a given set of output files should be retained or deleted automatically after successful transfer to

EDC DAAC, delete an output file set immediately, and enable/disable the automatic receipt of DDNs.

Figure 10–13. LPS GUI Display Operational Messages Dialog

Figure 10–14. LPS GUI Operational Messages Display Window

10.7.1 Resend DAN...

NOT IMPLEMENTED IN RELEASE 1

10.7.2 Retain File Set...

NOT IMPLEMENTED IN RELEASE 1

10.7.3 Delete File Set...

NOT IMPLEMENTED IN RELEASE 1

Figure 10–15. LPS GUI File Management Menu

10.7.4 Stop DDN Server...

Selecting this option causes the LPS software to stop the automatic receipt and processing of DDNs from EDC DAAC. The LPS GUI displays the Stop DDN Server Confirmation dialog (Figure 10–16) requesting the user to confirm the command. Clicking CANCEL cancels the operation; DDNs will be received and processed. Clicking OK terminates the DDN server process.

WARNING

For Release 1, stopping the DDN server while it is processing a DDN will cause it to terminate prematurely, possibly leaving the LPS in an inconsistent state.

Before shutting down the LPS software, you must verify that the DDN server is not currently processing a DDN by reviewing the LPS Journal file.

Figure 10–16. LPS GUI Stop DDN Server Confirmation Dialog

The command changes to **Start DDN Server…** whenever the DDN server has been stopped. When you select this option, the LPS GUI displays the Start DDN Server Confirmation dialog (Figure 10–17). Clicking CANCEL cancels the operation; DDNs will not be received and processed. Clicking YES activates the DDN server process. DDNs will be received and processed automatically.

10.8 Shutdown Menu

There is no pulldown menu for the Shutdown option. Clicking the mouse on this option will cause the LPS GUI to attempt to shut down the LPS software. The LPS GUI will display the confirmation dialog (Figure 10–18). Clicking CANCEL will cancel the operation, leaving the state of the LPS software unchanged. Clicking OK causes the LPS GUI to attempt the shutdown. The LPS GUI itself will exit and all LPS software automatic processes will be terminated.

Figure 10–17. LPS GUI Start DDN Server Confirmation Dialog

WARNING

For Release 1, shutting down the LPS software while the DDN server is processing a DDN will cause it to terminate prematurely, possibly leaving the LPS in an inconsistent state.

Before shutting down the LPS software, you must verify that the DDN server is not currently processing a DDN. See Appendix E, Section E.6.4, for instructions on determining whether the DDN server is active.

If the LPS software is currently capturing data or performing L0R processing, the shutdown command will display the shutdown failure dialog (Figure 10–19). Clicking OK is the only option. In this case, the state of the LPS software is unchanged.

Figure 10–18. LPS GUI Shutdown Command Confirmation Dialog

Figure 10–19. LPS GUI Shutdown Command Failure Dialog

Appendix B. Directory Structure and File Name Formats

This appendix describes the structure of the directories on LPS strings containing files of interest for LPS processing and the formats for LPS file names.

B.1 String Directory Structure

To be defined.

B.2 Output File Subdirectory Structure

LPS L0R processing software creates a new subdirectory in the directory pointed to by the LPS_OUTFILE_PATH environment variable for each instance of L0R processing. Within that directory, the software creates a separate directory for each subinterval extracted from the raw wideband data file. All output files generated during a particular L0R processing run are created in the directory for the subinterval to which they belong. The LPS_OUTFILE_PATH directory, therefore, has the structure shown in Figure B–1.

Contact Id is the contact ID assigned to the raw wideband data file being processed by the LPS database. *File Version No.* is the file version number assigned to the L0R processing run. *Subinterval Id* is the subinterval ID assigned to the subinterval by the LPS database.

B.3 Raw Wideband Data File Name Format

Raw wideband data file names have the following format:

YY-DDD-HH:MM:SS_String.cpt

YY is the last two digits of the year of capture. *DDD* is the Julian day of capture. *HH:MM:SS* is the time of capture (hour, minute, second) using a 24-hour clock. *String* is the host name (for example, lps001) of the LPS string that captured the data in the file.

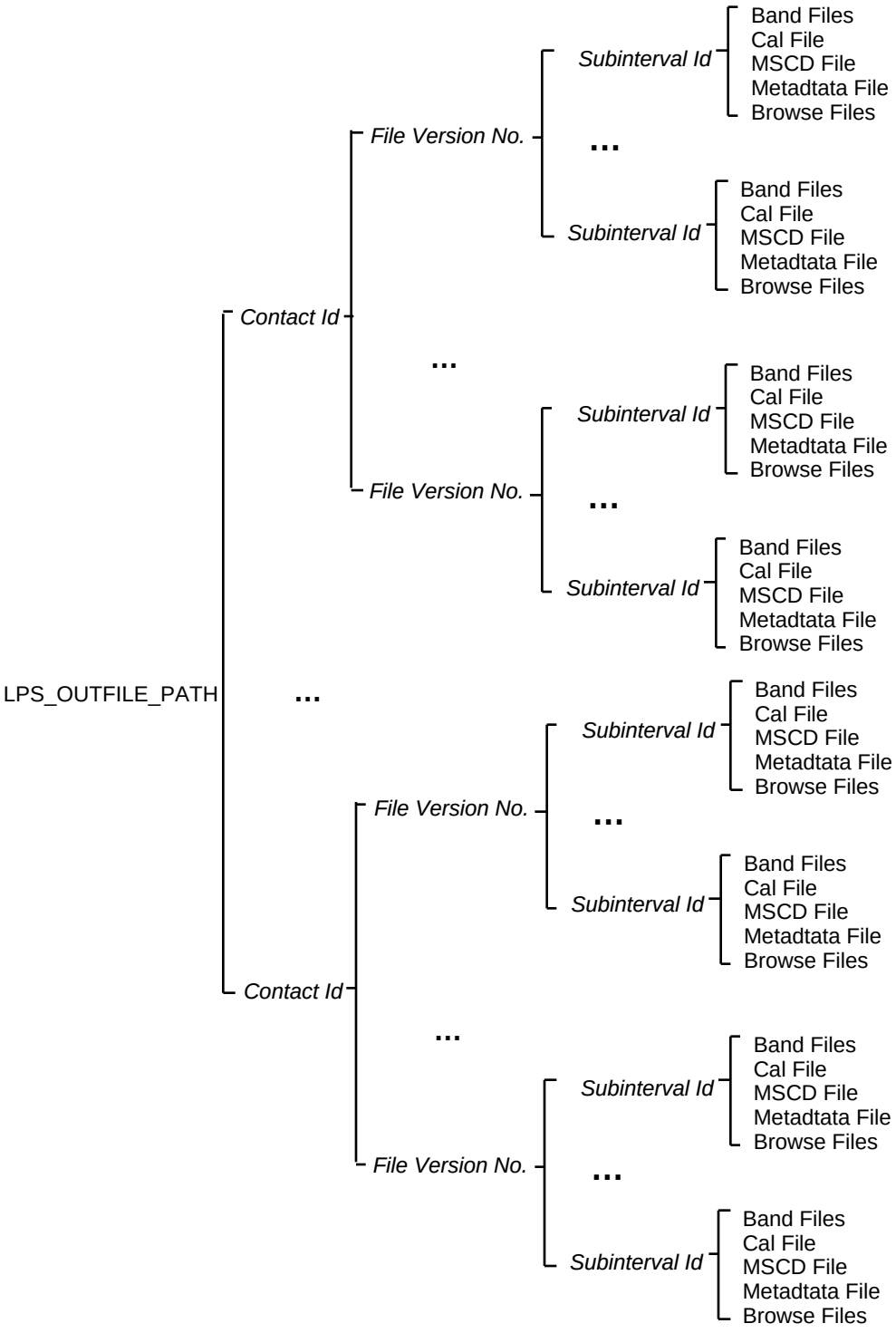

Figure B–1. LPS Output File Directory Hierarchy

B.4 L0R Output File Name Format

L0R output file names have the following format:

L7XsssfnYYDOYHHuuv.xxs.

X is the Landsat 7 X band (1, 2, or 3) on which the data was transmitted. *sss* is the data capture ground station ("EDC" for the EROS Data Center). *f* is the ETM+ format (1 or 2). *n* is the LPS processor number (the last character of the host name; lps001 = processor number 1, for example). *YY* is the last two digits of the year of capture. *DOY* is the Julian day of capture. *HH* is the hour of the capture time. *uu* is the sequence number of the subinterval among others in the contact (01 to 99). *v* is the file version number (0 to 9). *xxs* is the file type: B1, B2, B3, B4, B5, B6, B7, B8 for band files, MSD for MSCD files, PCD for PCD files, CAL for calibration files, MTA for metadata files, Rnn for browse files, nn = the sequence number of the scene within the subinterval (01 to 99).

B.5 Trouble File Name Format

L0R trouble file names have the following format:

*RawFileName_System*_TroubleFileV*FileVersionNumber*

RawFileName is the name of the raw wideband data file being processed (see Section B.4 for the raw file name's format). *System* is the LPS software subsystem that generated the file. If "RDPS" is the value for *System*, the trouble file contains transfer frames with quality annotations. If "MFPS" is the value for *System*, the trouble file contains sets of transfer frames with annotations plus additional annotations related to the attempt to produce an ETM+ major frame from them. *FileVersionNumber* is the file version number assigned to the instance of L0R processing that produced the file.

B.6 LPS/EDC DAAC Message File Name Formats

LPS/EDC DAAC messages are saved in files with names in the following format:

Type_NNNNNNNNNN_MM.msg

Type is the message type: DAN, DAA, DDN, or DDA. *NNNNNNNNNN* is the DAN sequence number associated with the set of messages. *MM* is the message number.

The format of the names of authentication messages is **TBD**.

Appendix C. Process Catalog

This appendix describes the function of each LPS software process. Descriptions are ordered by process name. The list includes not only LPS application software processes, but system software processes that have a significant role in performing LPS functions. The list does not include Forms 4.5 executables or interpreter scripts (such as those used by awk).

Each process is accompanied by an indication of whether or not it can be invoked as a standalone program from an IRIX shell. Further information on LPS standalone programs can be found in Appendixes E and F.

Process Name	Standalone?	Description
deletefile	TBA	NOT IMPLEMENTED IN RELEASE 1
filtermon	Y	Filters and formats messages as they are written to the LPS Journal file
genfts	TBA	NOT IMPLEMENTED IN RELEASE 1
idp	N	L0R process that controls the generation of band files and browse files, performs ACCA, and generates the MWD
idp_acca	TBA	NOT IMPLEMENTED IN RELEASE 1
idp_band	N	L0R process that generates band files
idp_browse	TBA	NOT IMPLEMENTED IN RELEASE 1
idp_mwd	TBA	NOT IMPLEMENTED IN RELEASE 1
mac_db_init	Y	Updates the online status of raw wideband data files in the LPS database by checking the raw data file storage array
mac_lps	Y	LPS startup process
mac_startl0r	Y	Controls L0R processing of a specified raw wideband data file
mfp	N	L0R process that constructs ETM+ frames and generates calibration and MSCD files
monsym	Y	Generates a formatted display of LPS status and error messages as they are written to the LPS Journal file
pcd	N	L0R process that constructs PCD frames, generates PCD files, and identifies WRS scene centers

Process Name	Standalone?	Description
rcvddn	Y	DDN server – accepts and processes DDNs from EDC DAAC; deletes successfully transferred files, notifies operator of file transfer errors, and updates output file transfer accounting in the LPS database
rdc	Y	Captures raw wideband data beginning immediately and continuing for a specified period of time
rdc_DeleteFiles	Y	Deletes a specified raw wideband data file and updates appropriate accounting information in the LPS database
rdc_GenReport	TBA	NOT IMPLEMENTED IN RELEASE 1
rdc_Restage	TBA	NOT IMPLEMENTED IN RELEASE 1
rdc_Save	TBA	NOT IMPLEMENTED IN RELEASE 1
rdc_ShutDown	Y	Stops raw wideband data capture, optionally after an specified period of time
rdc_StopSaveRestage	TBA	NOT IMPLEMENTED IN RELEASE 1
rdc_Terminate	Y	Script to terminate raw data capture (Build 1 only)
rdc_TestData	TBA	NOT IMPLEMENTED IN RELEASE 1
rdc_UpdAcct	TBA	NOT IMPLEMENTED IN RELEASE 1
rdp	N	L0R process that performs return-link processing to extract and validate transfer frames from raw wideband data
resenddan	TBA	NOT IMPLEMENTED IN RELEASE 1
retainfile	TBA	NOT IMPLEMENTED IN RELEASE 1
senddan	Y	Send a DAN to EDC DAAC for a specified L0R processing run
stopddn	Y	Stop the DDN server
syslogd	N	IRIX system log daemon process – accepts LPS status and error messages, formats and time tags them, and routes them to the LPS Journal file
xwsh	Y	IRIX X window shell program; invoked by monsym to produce a window to display LPS status and error messages
mac_autocapture	TBA	NOT IMPLEMENTED IN RELEASE 1

Appendix D. Customizing Your Environment

The user normally will not need to manipulate environment elements directly. The LPS .lpsrc file will automatically set up an environment (see Section 2.3). This appendix explains what environment variables are important for LPS operations and how to customize their values, should that be necessary. Specifically, this appendix covers the following topics:

- Setting up to use a different LPS software version
- Customizing environment variables within a particular version
- LPS-specific environment variables
- ORACLE environment variables
- Shell environment variables

NOTE

Your environments on each LPS string and operator interface workstation are independent. Any change must be made separately on each LPS string and workstation.

D.1 Setting Up to Use a Different LPS Software Version

To use a different version of LPS software, replace the definition of LPS_HOME at the top of your .cshrc file with the name of the top-level directory of the LPS software version you wish to use. Table D–1 lists the LPS software version currently installed on the LPS003 string.

Table D–1. LPS Software Version

Top-Level Path	Description
usr/LPS/at/r1	Release 1 acceptance test

D.2 Customizing Environment Variables Within a Particular Version

There are two basic approaches for customizing your environment:

1. Replace the standard .lpsrc file with your own custom version.
2. Override the standard environment variable values.

The first approach is effective when you will be changing the values of many environment. If you are making only a few changes, the second approach is probably more effective.

To replace the standard .lpsrc file with your own custom version, do the following on each LPS string and workstation:

1. Make a copy of the standard .lpsrc file in your home directory by typing:

   ```
   % cp $LPS_HOME/.lpsrc ~
   ```

2. Use a text editor to edit your copy of .lpsrc to change the values assigned to environment variables.

3. Use a text editor to edit your .cshrc file to change your sourcing of .lpsrc to the following:

   ```
   source ~/.lpsrc
   ```

To override standard environment variable settings, do the following on each LPS string and workstation:

1. Use a text editor to edit your .cshrc file.

2. After the "source $LPS_HOME/.lpsrc" line, add lines setting the environment variables you want to customize as follows:

   `setenv` *`Environment-Variable-Name Custom-Value`*

   For example, to override the standard settings of LPS_REPORT_PATH and LPS_TROUBLE_FILE_PATH to write reports and trouble files into two subdirectories of your home directory, you would add the following to your .cshrc file:

   ```
   setenv LPS_REPORT_PATH ~/myreports
   setenv LPS_TROUBLE_FILE_PATH ~/mytrouble
   ```

NOTE

Setting environment variables from the command line is not effective for commands and programs executed in subprocesses. The standard .lpsrc file is executed for every subprocess, resetting environment variables to their standard values. In particular, custom values will not be in effect when LPS software is activated unless the settings are made in your .cshrc file AFTER .lpsrc has executed.

D.3 LPS-Specific Environment Variables

The LPS software uses the environment variables listed in Table D–2. You can set these values to control the associated LPS behavior. Guidelines for setting values and restrictions on acceptable values appear in the descriptions.

D.4 ORACLE Environment Variables

The LPS software assumes that environment variables required by the ORACLE DBMS have been correctly defined. Table D–3 lists the ORACLE environment variables required for the LPS software.

For a detailed discussion of ORACLE environment settings consult *TBD ORACLE Reference Document* (Reference 6).

D.5 Shell Environment Variables

The LPS software makes use of the standard shell environment variables listed in Table D–4.

Table D–2. LPS Environment Variables (1 of 2)

Environment Variable	Description
LPS_BIN	Directory where the LPS software searches for executable images. This should be set to the path name of the directory containing the version of the LPS software you want to run. Normally set to $LPS_HOME/bin.
LPS_CAPTURE_PROCESSOR (**TBR**)	Processor number of processor to be used when the LPS data capture process, rdc, is specified to run on a restricted, isolated, and nonpreemptive processor running with the highest nondegrading priority.
LPS_DANFILE_PATH	Directory into which the LPS software writes a copy of each DAN it sends to EDC DAACs.
LPS_DDNFILE_PATH	Directory into which the LPS software write a copy of each DDN it receives from EDC DAAC.
LPS_HOME	Top-level directory containing the LPS software installation. To set up so that your environment points to a version of LPS software automatically, explicitly set this variable in your .cshrc file just prior to the “source $LPS_HOME/.lpsrc” line.
LPS_JOURNAL_PATH	Directory containing the LPS Journal file. WARNING: Changing the value of this variable does not change the directory in which the LPS Journal file resides. The location and name of the LPS Journal file is specified in the system file, /et/syslog.conf, as the destination for local0 facility messages. This variable should always be set to the same file.

Table D–2. LPS Environment Variables (2 of 2)

Environment Variable	Description
LPS_LOG_STDOUT	When set to any value, indicates that the LPS software should write status and error messages to stdout, as well as to the LPS Journal file. This variable should be set only when LPS software is being executed from a UNIX shell. WARNING: Running LPS software through the LPS GUI when this variable is set could hang the LPS software.
LPS_OUTFILE_PATH	Top-level directory for LPS output files. Should be set to the top-level directory of the file system on the LPS string's output redundant arrays of independent disks (RAID) device. Your user ID must have read permission for the directory.
LPS_RAWFILE_PATH	Directory into which LPS raw wideband data files will be written during data capture. Should be set to the top-level directory of the file system on the LPS string's input RAID device. Your user ID must have read permission for the directory.
LPS_REPORT_PATH	Directory into which the LPS software will write files containing LPS reports. Your user ID must have read and write permission for the directory.
LPS_TABLE_PATH	Directory holding files containing static data loaded by the LPS software during its execution. Normally, set to $LPS_HOME/tables.
LPS_TAPE_DEV	The name of the default tape device for copying raw wideband data files to the 30-day store.
LPS_TEMPFILE_PATH	Directory into which the LPS software will create temporary files. Normally set to /tmp. Your user ID must have read and write permission for the directory.
LPS_TROUBLEFILE_PATH	Directory into which the LPS software will write trouble files. Your user ID must have read and write permission for the directory.

Table D–3. ORACLE Environment Variables for LPS Software

Environment Variable	Description
FORMS45_PATH	Path name of directory containing LPS GUI Forms executables. Normally set to $LPS_HOME/bin
ORACLE_HOME	Path name of top-level directory of the ORACLE installation containing the bin, lib, guicommon, and formsXX subdirectories
ORACLE_PATH	List of directories containing executables; similar to the UNIX PATH variable
ORACLE_SID	Name of database instance to which LPS software will connect
ORACLE_TERM	Terminal type
TK2_TERMINAL	Terminal characteristics file

Table D–4. Shell Environment Variables

Environment Variable	Description
PATH	List of directories in which the shell will search for executables (see csh(1) for details of its operation. PATH should include both $LPS_HOME/bin and $ORACLE_HOME/bin.
MAN_PATH	List of directories containing man pages [see man(1) for details of its operation]. MAN_PATH should include $LPS_HOME/man for LPS man pages and appropriate directories in $ORACLE_HOME for ORACLE man pages.
DISPLAY	Address of X Window System display. DISPLAY should be set to "***Address***:0" where ***Address*** is the IP name or IP address of the X display device (X terminal or workstation).
LD_LIBRARY_PATH	Lists additional directories containing dynamic shared objects. The list should include the default directories (/lib & /usr/lib), as well as directories for ORACLE shared objects (consult Reference 6).

Appendix E. Performing LPS Functions Through IRIX

The LPS software is designed so that all major operations can be invoked from both the LPS GUI and any IRIX shell. Also, access to the LPS database is possible through both the LPS GUI and Oracle's SQL*Plus. This combination of capabilities allows the user to invoke LPS functions in ways not supported by the LPS GUI. In particular, it allows the user to create shell scripts that invoke LPS functions, thereby extending LPS capabilities.

NOTE

For Release 1, access to particular LPS functions, especially access to the LPS database, is not yet available through the LPS GUI. Performing the functions is possible only through IRIX or SQL*Plus, and instructions appear in the document's body. In these cases, this appendix contains only a reference to the section in the document.

E.1 Finding LPS Database Identifiers

The LPS software uses unique IDs to reference information about captured data and the output of L0R processing. To invoke LPS programs from IRIX, it is frequently necessary to supply these IDs. The following IDs are frequently needed:

- *Contact ID* – a number assigned automatically by the ORACLE DBMS to identify the data captured on a LPS string during a certain contact
- *File Version Number* – a number assigned by the operator to a particular L0R processing run
- *Subinterval ID* – a number assigned automatically by the ORACLE DBMS to identify a particular subinterval extracted from a raw wideband data file during a particular L0R processing run

For the IDs automatically assigned by the ORACLE DBMS, query the LPS database to determine what they are. For the file version number, you will frequently already know its value. However, it is possible to retrieve a list of file version numbers assigned to the processing runs on the string.

E.1.1 Finding a Contact Identifier

To find a contact ID, query the LPS database on the string storing the raw data capture file. You must know one of the following for the contact period:

- Scheduled capture start time
- Scheduled capture stop time
- Actual capture start time
- Actual capture stop time

- Raw data capture file name

Each LPS string assigns contact IDs to contacts independently. Raw wideband data files captured during the same contact by different strings may not have the same contact ID; files captured during different contacts may have the same IDs.

To retrieve the contact ID, type the following to SQL*Plus:

```
SQL> SELECT contact_sequence_id FROM rdc_acct WHERE condition;
```

The value you enter for *condition* will depend on what you know about the contact period. Table E–1 lists the values for *condition* for each of the items above. It also describes how to query for a list of contacts from which you can select the contact sequence ID.

Table E–1. Query Conditions for Selecting Contact Sequence IDs

If you know...	Then use this as the where clause
Scheduled capture start time	WHERE scheduled_start_time = TO_DATE('***Date&Time***', ***'Format'***) ***Format*** must describe the format of Date&Time. For example, 'YY-DDD-H24:MI:SS.' See Reference 6 for details on format options for the to_date() function.
Scheduled capture stop time	WHERE scheduled_stop_time = TO_DATE('***Date&Time***', ***'Format'***)
Actual capture start time	WHERE actual_start_time = TO_DATE('***Date&Time***', ***'Format'***)
Actual capture stop time	WHERE actual_stop_time = TO_DATE('***Date&Time***', ***'Format'***)
Raw data capture file name	WHERE raw_data_file_name = '***filename***'
Any field and want a list of all **unprocessed** contacts on the string's raw data capture storage	WHERE on_line_flag = 1 (**TBR**) AND contact_sequence_id not in (SELECT UNIQUE contact_sequence_id FROM processing_version_info) Add any combination of scheduled_start_time, scheduled_stop_time, actual_start_time, actual_stop_time, and raw_data_file_name to the list of selected attributes so that the contact can be identified. For example, "select contact_sequence_id, raw_data_file_name from..."
Any field and want a list of **all** contacts on the string's raw data capture storage	WHERE on_line_flag = 1 (**TBR**) Add any combination of scheduled_start_time, scheduled_stop_time, actual_start_time, actual_stop_time, and raw_data_file_name to the list of selected attributes so that the contact can be identified. For example, "select contact_sequence_id, raw_data_file_name from..."

E.1.2 Finding a File Version Number

Normally you will already know the file version number you want to use for a LPS operation. You can find a list of the file version numbers of L0R processing instances on this LPS string for a given contact by typing the following to SQL*Plus:

```
SQL> SELECT file_version_number FROM processing_version_info
2>   WHERE contact_sequence_id = Contact Identifier;
```

Contact Identifier is the contact ID for the contact.

Example: Retrieve a list of file version numbers for the contact with contact ID = 86214.

```
SQL> SELECT file_version_number FROM processing_version_info
2>   WHERE contact_sequence_id = 86214;
FILE_VERSION_NUMBER
-------------------
                  0
                  1
                  5
```

On this string, the contact ID with ID 86214 has L0R processing instances with file version numbers of 0, 1, and 5.

You can produce a list of all file version numbers together with other information describing the contact by typing the following to SQL*Plus:

```
SQL> SELECT file_version_number, scheduled_start_time,
scheduled_stop_time,
2>   actual_start_time, actual_stop_time, raw_data_file_name
3>   FROM processing_version_info, rdc_acct
4>   WHERE processing_version_info.contact_sequence_id =
rdc_acct.contact_sequence_id;
```

E.1.3 Finding a Subinterval Identifier

To find a subinterval ID you must know L0R processing instance that produced the subinterval, as well as either the subinterval's start or stop time or the subinterval's ordinal number within the L0R processing instance.

1. Find the contact ID and the file version number of the L0R processing instance that contains the subinterval. Sections E.1.1 and E.1.2 provide instructions on how to do so.
2. Look up the database attribute name of a known value identifying the subinterval in Table E–2.

Table E–2. Attribute Names for Subinterval Identifying Fields

Known Value	Attribute Name
Subinterval number within contact (1..35) (**TBR**)	sub_intv_number

Subinterval start time Format = YYYYDDDHHMMSSmmmxxxx YYYY = Year DDD = Julian day of year HH = Hours of day (24 hour clock) MM = Minutes of hour SS = seconds of minute mmm = milliseconds of second xxxx = 1/16 of millisecond	mf_start_time
Subinterval stop time Same format as start time	mf_stop_time

3. Type the following to SQL*Plus:

```
SQL> SELECT sub_intv_sequence_id FROM sub_intv
2>   WHERE contact_sequence_id = Contact Identifier
3>   AND file_version_number = File Version Number
4>   AND Attribute = Value;
```

Contact Identifier and *File Version Number* are the contact ID and file version number, respectively, you found in step 1. *Attribute* is the attribute name you found in step 2. *Value* is the value you know. Note that if you are using either subinterval time, you must type the time in the format described in Table E–2 and enclose the time in single quotes.

NOTE

If you know the subinterval's start or stop time in the format used in Landsat 7 metadata (YYYY:DDD:HH:MM:SS.tttttt), you must convert the time to the LPS database format. All fields are the same except for the fractions of seconds. To convert from metadata format,

mmm = the first three digits of the metadata format fractions of second

xxxx = tttt / 625, where tttt = the last four digits of the metadata format fractions of second

For example, to convert the fraction part of the metadata format time 1999:245:23:10:19.0163125,

mmm = the first three digits = 016

xxxx = 3,125/625 = 0005

So the database format time is 19992452310190160005.

Example: Find the subinterval ID for the subinterval with start time of day 210 of 1998 at 16:24:34.2946875 in the first instance of processing the contact scheduled to start on day 211 of 1998 at 09:07:36.

1. Using one of the methods described in Sections E.1.1 and E.1.2, the contact ID is discovered to be 86214 and the file version number is 0.
2. In Table E–2, the attribute containing the subinterval start time has the name "mf_start_time."
3. Type the following to SQL*Plus:

```
SQL> SELECT sub_intv_sequence_id FROM sub_intv
2>   WHERE contact_sequence_id = 86214
3>   AND file_version_number = 0
4>   AND mf_start_time = '19982101624342940l1';
SUB_INTV_SEQUENCE_ID
--------------------
               86325
1 row(s) selected.
```

The subinterval ID is 86325.

E.2 Status and Error Message Monitoring

This section describes how to monitor LPS status and error messages from IRIX. All LPS status and error messages are written to the LPS Journal file.

- Section E.2.1 describes the format of messages in the LPS Journal file.
- Section E.2.2 describes how to set up windows that display messages as they are written to the LPS Journal file. It also describes how to construct custom filters that will select messages with specific priorities and format the messages for more convenient display.
- Section E.2.3 describes how to set up your environment so that LPS status and error messages will be written in the window from which a LPS program was invoked.
- Section E.2.4 describes how to review the contents of the LPS Journal file using IRIX utilities such as pagers and editors.

E.2.1 Message Format

The LPS software uses the IRIX system log daemon, syslogd, to route status and error messages to the LPS Journal file. Most of the formatting for LPS messages is performed by syslogd. See syslogd(1M) in Reference 2 for details.

The following is an example of a line from the LPS Journal file:

```
Apr 2 15:36:22 1Q:lps001 syslog: 3475 [rdc_Main.c:208]
Suspending level 0R processes
```

The date and time of the message appear first, followed by the priority, facility, and hostname (for "1Q:lps001" in the example, '1' is the priority, 'Q' is the facility, and 'lps001' is the host). The source is the next item. For messages generated by the LPS software, the source is always "syslog." The process group ID of the process generating the message appears next ("3475" in the example). The unit name and source line appear next in square brackets. These are normally of interest only for software troubleshooting. The final part of the line is the message itself.

The priority is represented as a number between 0 and 7 using the standard IRIX priority scheme. Table E–3 summarizes the priorities and their meanings. Note that smaller numbers indicate higher priorities.

Table E–3. IRIX Message Priorities

Priority No.	Description
0	EMERGENCY – system is unusable
1	ALERT – immediate action must be taken
2	CRITICAL – critical condition
3	ERROR – error condition
4	WARNING – warning condition
5	NOTICE – normal but significant
6	INFO – informational message
7	DEBUG – debug level messages intended for software troubleshooting; not of operational interest

IRIX facility codes are represented as "A" through "T." The facility code for all LPS messages is "Q." This corresponds to the local0 facility [see logger(1) in Reference 2].

Normally, you will create a status/error message display window for each LPS string by selecting menu options from the LPS GUI. It is also possible to view LPS messages as they are written to the LPS Journal file from an IRIX shell.

E.2.2 Setting Up Message Displays

To view LPS software status and error messages in real time from a UNIX shell, type the following commands:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk ‘{print $NF}’`
% cat -f $LPSJournal
```

To view LPS software status and error messages in real time in a separate X window, type the following commands:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk ‘{print $NF}’`
% xwsh -autofork -hold -e cat -f $LPSJournal
```

See (1G) in Reference 2 for details on ways to customize the window.

By using standard IRIX filters, it is possible to customize your display. You can filter out low-priority messages and display only the fields of interest. See Section 2.7.1 for details of LPS message format.

Figure E–1 is an example of a custom filter. The awk program filters out DEBUG (7) and INFO (6) priority messages and displays only the time, priority (converted to a four-character mnemonic), and the message itself, excluding the unit name, line number, and process group ID.

If this program was stored in the file awk.script, typing the following would produce a custom-formatted real-time display in a separate window:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk ‘{print $NF}’`
% xwsh -autofork -hold \
? -e “tail -f $LPSJournal | awk -f awk.script”
```

Note the double quotes surrounding the argument to the -e option. They are necessary to force the shell to interpret “| awk -f awk.script” as part of the option rather than as directing that the output of xwsh be piped to awk.

```
BEGIN {
      priorities[0] = “EMRG”
      priorities[1] = “ALRT”
      priorities[2] = “CRIT”
      priorities[3] = “ERRO”
      priorities[4] = “WARN”
      priorities[5] = “NOTE”
}
substr($4,1,1) < 6 {
      printf “%s %s “, $3, priorities[substr($4,1,1)];
      for (i = 8; i <= NF; ++i)
        printf “%s “, $i;
      printf “\n”;
}
```

Figure E–1. Sample awk Script to Filter LPS Status and Display Messages

E.2.3 Directing Status and Error Message Display to the Standard Output

When running LPS programs directly from a shell rather than through the LPS GUI, you can direct the LPS software to write all status and error messages to the standard output (stdout) as well as to the LPS Journal file. This allows you to view the status and error messages from a process in the window from which it was invoked. The LPS program’s output can be piped to other programs to filter and format the output. It can also be redirected to a file using csh(1) input/output redirection commands.

To direct LPS messages to the standard output, set the value of LPS_LOG_STDOUT to 1.

WARNING

Processes invoked through the LPS GUI may hang when LPS_LOG_STDOUT is set. LPS_LOG_STDOUT should *never* be set in the shell from which the LPS GUI is invoked.

E.2.4 Browsing the LPS Journal

IRIX includes the sysmon(1M) utility for viewing system log files that can be used to browse the LPS Journal file. To browse the LPS Journal file using sysmon(1M), type the following:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk '{print $NF}'`
% sysmon -f $LPSJournal &
```

Information on the sysmon(1) utility is available in sysmon(1M) in Reference 2. The utility also provides online help.

The LPS Journal file is a text file and therefore can be browsed using any of the standard IRIX file display utilities, such as cat(1), more(1), and page(1). For example, to browse the LPS Journal file using more(1), type the following:

```
% set LPSJournal = \
? `grep local0.debug /etc/syslog.conf | awk '{print $NF}'`
% more $LPSJournal
```

The LPS Journal file can also be viewed using text editors. The file should not be locked or modified while the LPS software is running. To enter notations into the LPS Journal file while the LPS software is active [for example, to mark a change of shift, use the IRIX logger(1) command]. This avoids possible message corruption due to multiple, concurrent writers and ensures that message formats are correctly preserved.

E.3 Data Capture Operations

This section describes how to capture raw wideband data by invoking LPS capture software from an IRIX shell. This capability is intended for use during testing and for emergency situations in which data must be captured, but the LPS GUI and daemon processes are not functional (for example, when the ORACLE database server has failed).

- Section E.3.1 describes how to start data capture from IRIX.
- Section E.3.2 describes how to stop data capture from IRIX.
- Section E.3.3 describes how to restore LPS database consistency after capturing data when the ORACLE server is not running.
- Section E.3.4 describes how to set the capture source attribute in the LPS database directly through SQL*Plus.
- Section E.3.5 describes how to view and edit the contact schedule in the LPS database directly through SQL*Plus.

E.3.1 Starting Data Capture from an IRIX Shell

To capture data manually from an IRIX shell:

1. Create a new shell window or select an existing one on the LPS string that you want to capture data and move the cursor into the window's text area.

2. Type the following:

   ```
   % setenv LPS_LOG_STDOUT 1
   ```

   to direct status and error messages to the standard output (see Section 2.7.1.3). Messages will appear in this window.

   NOTE

   If a real-time message display is active (either created from the LPS GUI or a custom display you set up yourself as described in Section 2.7.1), capture status and error messages will appear there.

3. Invoke the LPS capture program, rdc, by typing the following command:

   `% rdc -l` *`Capture_Source_ID`* `[-b` *`Start-Time`*`][-e` *`Seconds`*`][-s][-i]`

   The `rdc -l` *`Capture_Source_ID`* command is the minimum required to capture data.

   WARNING

   The -l option is mandatory. The rdc program will not run unless a value is supplied. If the correct value for the LGS channel is unknown, supply a valid dummy value (any number between 1 and 5) and correct it later in the LPS database.

   The remaining -b and -e options should be supplied if you know the correct values. The -b option indicates the scheduled start time for the contact being captured. The format should be "YYDDDHHMM:SS." The -e option indicates the length of time, in seconds, that rdc should capture data. The -s option causes rdc to attempt to suspend L0R processing before capturing data. The -i option causes rdc to attempt to run on a restricted, isolated, and nonpreemptive processor with highest nondegrading priority.

   For more details on rdc and its parameters, consult the man page for rdc(1lps), available online and in Appendix C.

   NOTE

   The -s and -i options are not implemented in Build 1.

4. Monitor messages from rdc to verify that capture is active.

E.3.2 Stopping Data Capture From an IRIX Shell

To stop data capture from an IRIX shell:

1. Create a new shell window or select an existing one on the LPS string that you want to capture data and move the cursor into the window's text area.
2. Type the following:

   ```
   % rdc_Terminate
   ```

3. Monitor messages from rdc to verify that capture has terminated. You should see a message that resembles

   ```
   Apr 2 15:36:22 1Q:lps001 syslog: 3475 [rdc_Main.c:208] Shutting
   down raw data capture
   ```

   If you invoked capture from an IRIX shell and set the LPS_LOG_STDOUT environment variable, the message will appear in the window in which you executed rdc. In any case, the message will appear in the LPS Journal file. For instructions on monitoring the LPS Journal file, see Section 2.7.1.

E.3.3 Cleaning Up After Capturing From an IRIX Shell

For Release 1, the LPS software does not insert raw data accounting information into the database. Therefore, no action is necessary to maintain LPS database consistency after data is captured when the ORACLE server is not running.

E.3.4 Setting the Capture Source From ORACLE SQL*Plus

See Section 3.2.6.

E.3.5 Updating the Contact Schedule From ORACLE SQL*Plus

NOT IMPLEMENTED IN RELEASE 1

E.4 Raw Wideband Data Management

NOT IMPLEMENTED IN RELEASE 1

E.5 L0R Processing Management

This section describes how to invoke L0R processing from an IRIX shell.

- Section E.5.1 describes how to start L0R processing from IRIX.
- Section E.5.2 describes how to stop L0R processing from IRIX.

E.5.1 Starting L0R Processing

To perform L0R processing manually from an IRIX shell:

1. Identify the contact ID of the contact period you want to process. See Section E.1.1 for instructions.

2. Identify the file version number for the processing run you are going to start. For contacts that have never been processed, the file version number should be 0. The file version number should increment by 1 each time the contact is reprocessed. If you do not know the file version numbers that have been assigned to a contact, Section E.1.2 provides instructions to retrieve them from the LPS database.

WARNING

The LPS software will fail with an error if the file version number you supply has already been used on this LPS string. It will process the contact with the file version number you supply even if you skip a file version number or if the file version number has already been used on another LPS string. Consult *EDC Landsat 7 DHF Standard Operating Procedures* (Reference 7) for instructions for determining the file version number.

3. Verify that the LPS instance of the ORACLE DBMS server process is active by typing:

```
% ps -ef | grep ora | grep -v grep
```

If the ORACLE DBMS server process instance is active, the output from the command will resemble the following:

```
oracle 24628   1 0  May 20 ?    0:42 ora_d000_LPS
oracle 24623   1 0  May 20 ?    4:57 ora_dbwr_LPS
oracle 24627   1 0  May 20 ?    0:15 ora_s000_LPS
oracle 24626   1 0  May 20 ?    0:03 ora_reco_LPS
oracle 24624   1 0  May 20 ?    3:48 ora_lgwr_LPS
oracle 24622   1 0  May 20 ?    3:21 ora_pmon_LPS
oracle 24625   1 0  May 20 ?    0:41 ora_smon_LPS
```

If the command does not produce a list of process names, the ORACLE server instance for the LPS is not running and the LPS software cannot perform L0R processing. Contact the LPS system administrator and do not proceed until the ORACLE server instance for the LPS is active.

4. Start L0R processing by typing the following:

   ```
   % mac_startl0r CSID Version [0 | 1]
   ```

   Use "1" as the last parameter if you want the LPS software to delete the raw data capture file when L0R processing completes successfully. Use "0" as the last parameter if you want to retain the file online after L0R processing completes successfully.

   NOTE

   The LPS software will not delete a raw data capture file if it is stopped manually or if an error causes processing to terminate abnormally.

E.5.2 Stopping L0R Processing

To stop L0R processing from an IRIX shell:

1. Determine the process group ID for the L0R processing you invoked by typing the following command:

   ```
   % ps -ef | grep mac_startl0R | grep -v grep
   ```

   The output will be a line that resembles the following:

   ```
   lps_op 5206 4602 0 17:51:52 pts/0 0:00 mac_startl0r 2354766 1 0
   ```

   The process group ID is the second item on the line (5206 in the example):

   NOTE

   If multiple L0R processes are executing on the LPS string, the command in step 1 will produce a line for each instance of L0R processing. The last item in the line is the command you used to invoke L0R processing. You can identify the process group ID of the L0R processing of interest by finding the contact sequence ID of the contact (appearing immediately after "mac_startl0r").

   You can also determine the process group ID from the status and error messages output by the L0R process. The process group ID appears after the source and before the unit name. For example, in the message `Apr 2 15:36:22 1Q:lps001 syslog: 3475 [mfp_Main.c:208] CAL file could not be opened`, 3475 is the process group ID.

2. In an IRIX shell, type the following:

   ```
   % mac_stopl0r process group ID
   ```

   The L0R process will output a series of status and error messages that can be monitored to verify that the shutdown has occurred (see Section 5.4).

E.5.3 L0R Processing Parameters Management

See Section 5.5.

E.5.4 L0R Error Reporting Management

See Section 5.6.

E.6 File Transfers to EDC DAAC

This section describes how to control LPS file transfers to EDC DAAC through an IRIX shell or SQL*Plus.

- Section E.6.1 describes how to access LPS/EDC DAAC communications parameters through SQL*Plus.
- Section E.6.2 describes how to set LPS/EDC DAAC communications parameters through SQL*Plus.
- Section E.6.3 describes how to control automatic transmission of DANs to EDC DAAC through SQL*Plus.
- Section E.6.4 describes how to determine the status of the DDN server through IRIX.
- Section E.6.5 describes how to stop and then reactivate the DDN server through IRIX.
- Section E.6.6 describes how to check whether EDC DAAC has been notified.
- Section E.6.7 describes how to determine the transfer status of a set of L0R output files through SQL*Plus.
- Section E.6.8 describes how to determine through IRIX whether EDC DAAC is the process of transferring files.
- Section E.6.9 describes how to access the retention status of a file set through SQL*Plus.
- Section E.6.10 describes how to mark file sets for retention and how to delete output files through IRIX and SQL*Plus.
- Section E.6.11 describes how to delete L0R output files.

E.6.1 Viewing LPS/EDC DAAC Communication Parameters

See Section 6.2.1.

E.6.2 Setting LPS/EDC DAAC Communication Parameters

See Section 6.2.2.

E.6.3 Automatic File Availability Notification Management

See Section 6.3.

E.6.4 Checking Whether the DDN Server is Active

To determine whether the DDN server is active, type the following to an IRIX shell:

```
% ps -ef | grep rcvddn | grep -v rcvddn
```

If the DDN server is active, this command will output a line that looks like

```
30 S 3098 5992 5991 0 36 24 * 3575:924 8f60e434 ttyd1 rcvddn
```

The command generates no output if the DDN server is not active.

E.6.5 Stopping the DDN Server

To stop the DDN server, type the following to an IRIX shell:

```
% killall -TERM rcvddn
```

The killall command produces no output. Follow the instructions in Section E.6.4 to verify that you have successfully stopped the DDN server.

E.6.6 Checking Whether EDC DAAC Has Been Notified

See Section 6.5.1.

E.6.7 Checking Whether EDC DAAC Has Transferred Files

See Section 6.5.2.

E.6.8 Checking Whether EDC DAAC is Transferring Files

See Section 6.6.

E.6.9 Marking a File Set for Retention

See Section 6.8.1.

E.6.10 Determining Whether a File Set Has Been Marked for Retention

See Section 6.8.2.

E.6.11 Deleting L0R Output Files

See Section 6.9.

E.7 Report Generation

NOT IMPLEMENTED IN RELEASE 1

E.8 Testing the LPS

NOT IMPLEMENTED IN RELEASE 1

Appendix F. Man Pages for LPS Standalone Programs

NOT IMPLEMENTED IN RELEASE 1

Abbreviations and Acronyms

ACCA	Automated Cloud Cover Assessment
ASCII	American Standard Code for Information Interchange
BCH	Bose-Chaudhuri-Hocquenghem (error detection and correction)
CADU	channel access data unit
CCA	Cloud Cover Assessment
CCR	configuration change request
CCSDS	Consultative Committee on Space Data Systems
CNMOS	Consolidated Network Management and Operations Support
CRB	Central Review Board
CRC	cyclic redundancy code
DAAC	Distributed Active Archive Center
DAA	data availability acknowledgment
DAN	data availability notice
DBA	database administrator
DBMS	database management system
DCN	document change notice
DDA	data delivery acknowledgment
DDN	data delivery notice
DLT™	Digital Linear Tape
ECS	EOS Core System
EDC	EROS Data Center
ETM+	Enhanced Thematic Mapper Plus
EROS	Earth Resources Observation System
FDDI	fiber-distributed data interface
FTP	File Transfer Protocol
GB	gigabyte
GTSIM	Generic Telemetry Simulator
GUI	graphical user interface

IAS	Image Assessment System
ID	identifier
IFOV	instantaneous field-of-view
IP	Internet Protocol
L0R	Level Zero R
LGS	Landsat Ground System
LPS	Landsat 7 Processing System
Mbps	megabits per second
MB	megabyte
MOC	Mission Operations Center
MSCD	mirror scan correction data
MWD	Moving Window Display
PCD	payload correction data
Q&A	quality and accounting
RAID	redundant arrays of independent disks
RAM	random access memory
SCID	spacecraft identifier
SGI	Silicon Graphics, Inc.
SQL	Structured Query Language
TBA	to be added
TBD	to be defined
TBR	to be resolved
UTC	universal time coordinated
WRS	Worldwide Reference System

Glossary

Band file — A **Level 0R output file** containing the image data from a single band in a single subinterval.

Browse image file — A **Level 0R output file** containing a reduced resolution image in three spectral bands of a scene.

Calibration file — A **Level 0R output file containing** the calibration data collected at the end of each scan for a single subinterval.

Contact (period) — The time period between the start and end of wideband data transmissions from the Landsat 7 satellite to a ground station.

Daemon (process) — A process without a controlling terminal that executes in the background. LPS daemon processes perform automatic **data capture**, copying to tape, **Level 0R processing**, and **DDN** processing. The IRIX **syslogd** daemon formats and routes LPS status and error messages to the **LPS Journal file**.

Data availability acknowledgment (DAA)

An LPS/EDC DAAC communication message sent by EDC DAAC to acknowledge receipt of a **data availability notice**.

Data availability notice (DAN)

An LPS/EDC DAAC communication message sent by the LPS to notify EDC DAAC that a **file set** is available for transfer.

Data delivery acknowledgment (DDA)

An LPS/EDC DAAC communication message sent by the LPS to acknowledge receipt of a **data delivery notice**.

Data delivery notice (DDN)

An LPS/EDC DAAC communication message sent by EDC DAAC to notify the LPS of the disposition of its attempts to transfer a **file set**.

DDN server — An LPS **daemon** that accepts and processes **DDNs** from EDC DAAC.

Enhanced Thematic Mapper Plus (ETM+)

The imaging instrument onboard Landsat 7.

ETM+ format — One of two 150-Mbps telemetry streams output by the ETM+. Format 1 contains spectral bands 1 through 6. Format 2 contains spectral bands 6, 7, and 8. Both formats include calibration data , mirror scan correction data, and payload correction data.

File group — A collection of **Level 0R output files** for a single subinterval comprising one component of **file set**.

File set	A collection of all **Level 0R output files** for a single **Level 0R processing** run. The file set includes **file groups**, one for each subinterval extracted by the processing.
I channel	A 75-Mbps portion of an **X-band** channel containing either **format 1** or **format 2** telemetry data.
IRIX	A version of the UNIX operating system running on **LPS strings** and Indy **workstations**.
Level 0R output files	The set of files produced by L0R processing, consisting of **band files**, **browse files**, **calibration files**, **MSCD files**, and **metadata files.**
Level 0R processing	The operation of extracting Landsat 7 images, as well as correction and calibration information from **raw wideband data**, to produce **Level 0R output files**.
Level 0R Q&A data	The data Q&A information collected by the LPS from processing of the ETM+ major frames during **Level 0R processing**.
LPS database	A persistent storage repository containing configuration elements that can be set, **Level 0R processing** parameters and error reporting thresholds as well as Q&A information for **data capture**, **Level 0R processing**, and **Level 0R output file** transfer.
LPS GUI	The LPS graphical user interface that provides access to LPS functions on a single **LPS string**. The LPS GUI consists of a main menu from which LPS commands can be selected, additional dialogs for providing parameters to the command, and confirmation dialogs to prevent accidental command execution.
LPS Journal file	An ASCII file containing all of the status and error messages generated by the LPS software.
LPS string	One of five computers that hosts LPS processing. Four LPS strings are operational and perform **data capture** (each captures a single 75-Mbps channel) and **Level 0R processing** functions. A fifth string is used for training and development and as an emergency backup.
Metadata file	A **Level 0R output file** containing information on the L0R data provided in the subinterval and the names of the L0R **band file**, **calibration file**, **PCD file**, **MSCD file,** and **browse files** associated with the subinterval. Metadata also contains Q&A information on the data contained in the subinterval.

Mirror scan correction data (MSCD) file

A **Level 0R output file** containing the first half scan error, second half scan error, and scan direction for each scan.

ORACLE — A commercial DBMS used to manage the **LPS database** on each **LPS string**.

Payload correction data (PCD) file

A **Level 0R output file** containing PCD major frames received during a subinterval.

Playback (telemetry) — Telemetry data that is stored onboard the satellite when it is generated and replayed to a ground station at a later time.

Q channel — A 75-Mbps portion of an **X-band** channel containing either **format 1** or **format 2** telemetry data.

Raw wideband data — Telemetry data as received from Landsat 7 through the Landsat Ground Station.

Real-time (telemetry) — Telemetry data that is transmitted to a ground station as it is generated by the ETM+.

Retention status — Whether or not a **Level 0R output file** is marked in the **LPS database** as to be deleted automatically after successful transfer to EDC DAAC.

Return-link Q&A data

The data Q&A information collected by LPS L0R processing from CCSDS Grade 3 and BCH error detection and correction processing during **L0R processing**.

Shell — One of several IRIX programs that provide a command line interface for invoking programs, built-in commands for performing a set of operations such as echoing input, and an environment in which variables controlling program operation can be assigned persistent values. Shells also understand rudimentary flow control and will read script files containing commands. This provides a programming capability. Popular shells include the C shell [csh(1)] and Bourne shell [sh(1)].

SQL*Plus — The **ORACLE** SQL interpreter.

Subinterval — A segment of continuous **raw wideband data** received during a Landsat 7 **contact period**. Subintervals are caused by a shift from real-time to playback data or by a shift from one segment of playback to another collected at different time and location. Subintervals may also occur because of breaks in the wideband data stream due to communication dropouts. The largest possible subinterval can be as long as a full contact. The smallest possible subinterval can be as small as one full **WRS scene** with a duration of approximately 24 seconds.

telnet — An IRIX program that allows login and interaction with a remote host.

Workstation	An LPS hardware component used to host the **Moving Window Display** and, optionally, providing an interface to the **LPS strings**. Two workstations are normally used to host the **Moving Window Display** with each workstation hosting the display for two **LPS strings**. A third workstation is used for training and development and as an emergency backup.
WRS Scene	A frame of imagery defined by the Worldwide Reference System. Each scene is defined as a swath of approximately 179 kilometers (163 kilometers plus 10 percent in-track overlap) centered on a position along the Landsat 7 satellite's track. The WRS assigns sequential path numbers to the satellite's 251 nominal orbit tracks. The WRS also defines 119 numbered latitudinal center lines, called rows, that intersect the paths (row 60 is at the equator). WRS scene centers are defined as the points of intersection of the path and row lines. The scenes are designated by the path and row number at the scene center.
X band	A 150-Mbps downlink channel from Landsat 7. The satellite has three X-band channels. During any contact, at most two of them are active. Each X-band downlink consists of two 75-Mbps **I** and **Q channels.**
X terminal	An LPS hardware component providing an interface to the **LPS strings** and **workstations**. Two X terminals normally host the **LPS GUIs** for the four operational strings, with each X terminal hosting two strings. A third X terminal is used for training and development and as an emergency back-up.
xwsh	An IRIX program that provides an X window interface to a **shell** or to a specified application program.

References

1. *LPS Programmers Reference Manual*, publishing information TBA
2. *IRIX 5.3 Reference Manual*, publishing information TBA
3. *LPS Operations and Maintenance Manual*, 514-3OMM/0196, November 27, 1996
4. *LPS Installation Procedure*, 514-2IP/0195, October 14, 1996
5. *IRIX Essentials*, publishing information TBA
6. *TBD ORACLE Reference Document*, publishing information TBA
7. *EDC Landsat 7 DHF Standard Operating Procedures*, publishing information TBA

Index

TO BE ADDED